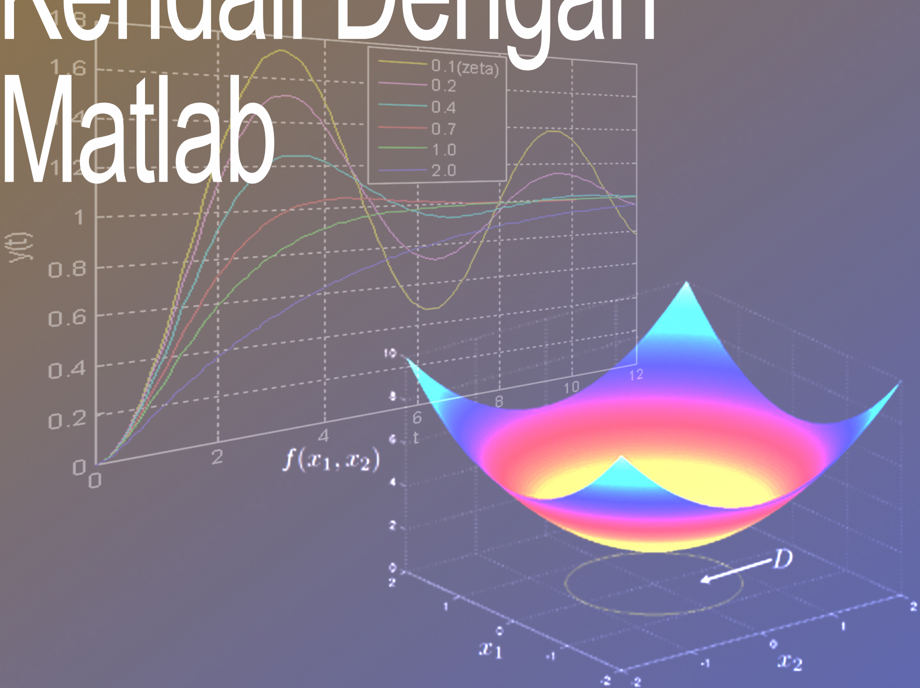
Heru Dibyo Laksono

Metoda –metoda Untuk Analisa Kestabilan Sistem Kendali Dengan Matlab



ISBN: 978-602-5539-23-7

METODA –METODA UNTUK ANALISA KESTABILAN SISTEM KENDALI DENGAN MATLAB

Heru Dibyo Laksono

Lembaga Pengembangan Teknologi Informasi dan Komunikasi
(LPTIK)

Metoda –Metoda Untuk Analisa Kestabilan Sistem Kendali Dengan Matlab

Penulis: Heru Dibyo Laksono

Tata Letak: Multimedia LPTIK
Sampul : Multimedia LPTIK

ISBN: 978-602-5539-23-7

Diterbitkan oleh:

Lembaga Pengembangan Teknologi Informasi dan Komunikasi (LPTIK) Universitas Andalas Lantai Dasar Gedung Perpustakaan Pusat Kampus Universitas Andalas Jl. Dr. Mohammad Hatta Limau Manis, Padang, Sumatera Barat, Indonesia
Web: www. lptik.unand.ac.id
Telp. 0751-775827 - 777049
Email: sekretariat_lptik@unand.ac.id

PRAKATA

Buku ini digunakan sebagai penunjang mata kuliah sistem kendali. Penekanan utama yang diberikan pada buku ini adalah analisa kestabilan pada sistem kendali dengan menggunakan berbagai metoda dan perangkat lunak Matlab. Adapun metoda – metoda yang dibahas untuk analisa kestabilan sistem kendali diantaranya persamaan karakteristik, kriteria Routh, kriteria Hurwitz, kriteria *Continued Fraction*, kriteria *Nyquist*, kriteria *Nichols*, kriteria Bode, metoda tempat kedudukan akar dan metoda Lyapunov kedua.

Ucapan terima kasih dan penghargaan yang setinggi-tingginya penulis sampaikan kepada Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) Universitas Andalas yang telah memberikan hibah untuk penerbitan naskah buku ini. Ucapan terima kasih kami sampaikan juga kepada orang tua dan banyak pihak yang telah memberikan perhatian penuh dalam penyelesaian naskah buku ini. Anak - anakku Thanisa Nashwa Azura (Thata) dan Fathan Athallah Kaysan (Fathan) serta keluarga besarku, buku ini kupersembahkan untuk kalian semua. Akhirnya, segala tanggungjawab akademis dari naskah buku ini sepenuhnya berada di tangan penulis.

Padang, April 2017

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I.
TINJAUAN UMUM

1.1 Penjelasan Umum

Buku ini membahas tentang penggunaan perangkat lunak Matlab untuk membantu penyelesaian - penyelesaian perhitungan metoda numerik. Pada saat ini komputer digital dan metoda numerik telah mengubah cara menghitung dan menyelesaikan masalah dengan lebih cepat dan lebih efisen. Namun demikian peran manusia sebagai perumus kreatif harus dilakukan untuk membuat masalah menjadi sederhana dan bermanfaat. Metoda numerik merupakan suatu cara yang mempuyai kemampuan sangat baik. Cara ini dapat digunakan untuk menyelesaikan sejumlah besar persamaan, sistem non linear dan bentuk geometri yang tidak umum. Hal ini biasa terdapat pada masalah teknik dan sains namun tidak ada solusi analitiknya. Dengan metoda numerik manusia akan terbebas dari hitung – menghitung manual yang membosankan oleh karena itu waktu dapat lebih banyak digunakan untuk tujuan yang lebih kreatif seperti penekanan pada formulasi problem dan interprestasi solusi. Pengetahuan metode numerik sangat diperlukan untuk bisa mengetahui fungsi kerja perangkat lunak tertentu dan salah satunya perangkat lunak Matlab. Dalam kasus – kasus tertentu perangkat lunak Matlab bahkan dapat digunakan untuk membuat perangkat lunak sendiri.

1.2 Tujuan

Setelah membaca dan mempelajari buku ini diharapkan

a. Pembaca dapat memahami dasar – dasar pemograman dengan menggunakan perangkat lunak Matlab
b. Pembaca dapat memnyelesaikan perhitungan – perhitungan metoda numerik dengan menggunakan bantuan Matlab

1.3 Buku Penunjang

Adapun buku penunjang yang bisa digunakan adalah

a. Chapra, S. C., & Canale , R. P. (2007). *Metode Numerik Untuk Teknik Dengan Penerapan Pada Komputer Pribadi* . Jakarta : Universitas Indonesia Press .
b. Munir, R. (2010). *Metode Numerik Revisi Ketiga* . Bandung : Informatika
c. Pujiyanta, A. (2007). *Komputasi Numerik denga Matlab* . Jogjakarta: Graha Ilmu.

1.4 Gambaran Umum

Buku ini terdiri dari 11 Bab. Antara satu bab dengan bab lain merupakan satu kesatuan rangkaian pembahasan yang utuh.

Bab I membahas tentang buku ini secara umum. Beberapa informasi dalam bab I ini akan membuat pembaca mengerti tujuan dari penulisan buku ini.

Bab II membicarakan tentang fasilitas – fasilitas yang terdapat pada Matlab. Materi yang dibahas meliputi window – window pada Matlab, komentar dan tanda baca pada Matlab, Matlab dan matematika sederhana, fungsi – fungsi dasar pada Matlab seperti fungsi – fungsi matematika dasar, fungsi – fungsi trigonometri dan fungsi – fungsi analisis data.

Bab III menjelaskan tentang variabel dan tipe data pada Matlab. Materi yang dibahas meliputi variabel, string, skalar, array sel, struktur array dan array multidimensi.

Bab IV diawali dengan pembahasan vektor kemudian dilanjutkan dengan pembahasan matrik pada Matlab. Untuk vektor materi yang dibahas meliputi notasi titik dua, vektor kolom, transportasi, perkalian, pembagian, pangkat dan ekstraksi bagian suatu vektor. Untuk matrik, materi yang dibahas meliputi matrik khusus, membuat matrik, ekstrak bagian matrik, operasi dan fungsi pada matrik.

Bab V membahas struktur kontrol pada Matlab. Materi yang dibahas meliputi operator relasi dan logika pada Matlab, perintah *if*, perintah *switch*, perintah *for*, perintah *while*, perintah *continue*, perintah *break* dan perintah *return*.

Bab VI mendiskusikan tentang pembacaan dan penulisan data pada Matlab. Materi yang dibahas meliputi pembacaan dan penulisan data dalam bentuk file .mat, pembacaan dan penulisan data dalam bentuk file .txt serta pembacaan dan penulisan data dalam bentuk file .xls.

Bab VII menjelaskan tentang visualisasi Matlab. Materi yang dibahas meliputi visualisasi gambar 2 dimensi, visualisasi gambar 3 dimensi dan visualisasi beberapa fungsi dalam satu gambar.

Bab VIII membicarakan tentang akar – akar persamaan nonlinier. Materi yang dibahas terbagi atas 2 bahagian yaitu metoda tertutup dan metoda terbuka. Untuk metoda tertutup terdiri dari metoda grafis, metoda bagi dua dan metoda posisi palsu. Untuk metoda terbuka terdiri dari metoda iterasi satu titik sederhana, metoda Newton Raphson dan metoda Secant.

Bab IX diawali dengan pembahasan pengertian sistem persamaan linier, jenis – jenis matrik dan operasi baris elementer. Untuk metoda – metoda dalam penyelesaian sistem persamaan linier yang dibahas diantaranya metoda eliminasi Gauss, metoda Gauss Jordan, metoda Gauss Seidel dan metoda dekomposisi LU. Untuk metoda dekomposisi LU ini terbagi atas 3 algoritma diantaranya algoritma Crout, algoritma Doolitle dan algoritma Cholensky.

Bab X membahas tentang integral numerik. Beberapa metoda yang dibahas diantaranya integral numerik dengan aturan Riemann, integral numerik dengan titik tengah, integral numerik dengan aturan trapezoid dan integral numerik dengan aturan Simpson.

Bab XI mendiskusikan tentang turunan numerik. Adapun metoda yang dibahas pada bahagian ini terdiri dari turunan numerik dengan metoda Euler dan turunan numerik dengan metoda Runge – Kutta.

BAB II
MATLAB

2.1 Pendahuluan

Pada bagian ini dibahas tentang Matlab. Pembahasan tentang Matlab ini terdiri dari variabel Matlab, ruang kerja Matlab, komentar dan tanda baca Matlab. Pembahasan dilanjutkan tentang Matlab dan matematika sederhana serta fungsi dasar pada Matlab. Untuk fungsi – fungsi dasar pada Matlab terdiri dari fungsi matematika dasar, fungsi trigonometri dan fungsi analisis data. Pembahasan diakhiri dengan rangkuman dan soal – soal

2.2 Matlab

Matlab adalah sebuah program untuk analisis dan komputasi numerik serta merupakan suatu bahasa pemograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matrik. Pada awalnya, program ini merupakan interface untuk koleksi rutin- rutin numerik proyek LINPACK dan EISPACK dan dikembangkan menggunakan bahasa Fortran. Namun sekarang program ini merupakan produk komersial dari perusahaan Mathworks, Inc. yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan Assembler. Matlab telah berkembang menjadi sebuah *environment* pemograman yang canggih dan berisi fungsi – fungsi *built-in* untuk melakukan tugas pengolahan sinyal, aljabar linier dan kalkulasi matematis lainnya. Matlab juga berisi toolbox yang berisi fungsi – fungsi tambahan untuk aplikasi khusus. Matlab juga bersifat *extensible* , dalam arti bahwa seorang pengguna dapt menulis fungsi baru untuk ditambahkan di *library* jika fungsi – fungsi *built-in* yang tersedia tidak dapat melakukan tugas tertentu. Kemampuan pemograman yang dibutuhkan tidak terlalu sulit bila pembaca telah memiliki pengalaman dalam bahasa pemograman bahasa lain seperti C, Pascal atau Fortran. Matlab merupakan bahasa pemograman tingkat tinggi berbasis pada matriks sering digunakan untuk teknik komputasi dan digunakan untuk menyelesaikan masalah – masalah yang melibatkan operasi matematika, elemen matrik, optimasi, aproksimasi dan lain – lain. Selain itu Matlab banyak digunakan untuk

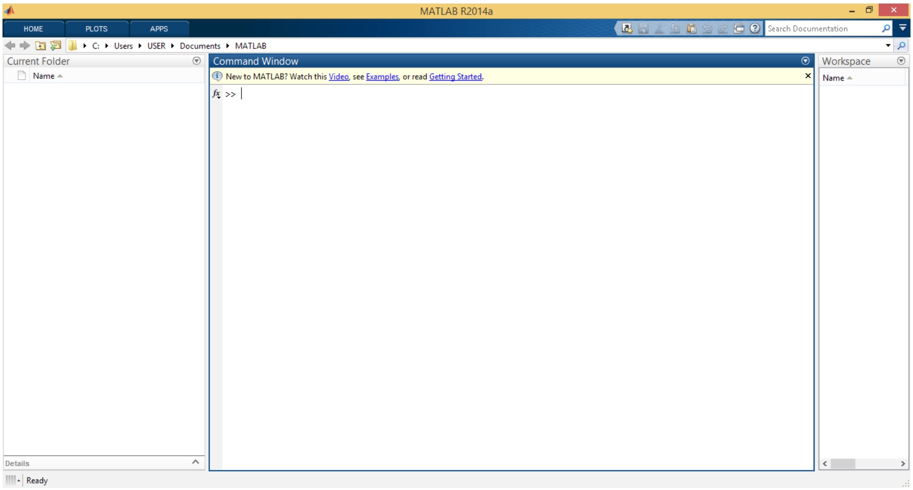
- Matematika dan komputasi.
- Pengembangan dan algoritma.
- Pemograman pemodelan, simulasi dan pembuatan prototipe.
- Analisa data, eksplorasi dan visualisasi
- Analisa numerik dan statistik
- Pengembangan aplikasi teknik

2.3 Window – Window Pada Matlab

Beberapa macam window yang tersedia dalam Matlab sebagai berikut

a. Matlab Command Window/Editor

Matlab Command Window/Editor merupakan window yang dibuka pertama kali setiap Matlab dijalankan pertama kali. Tampilan diperlihatkan pada Gambar 2.1 berikut



Gambar 2.1 Tampilan Matlab Setiap Kali Dijalankan

Pada window yang diperlihatkan pada Gambar 2.1 dapat dilakukan akses ke *command – command* Matlab dengan mengetikan barisan – barisan ekspresi Matlab seperti akses *help window* dan lain – lainya. Jika perintah – perintah yang sudah diketikan dan hasil yang ditampilkan pada layar *command window* akan disimpan maka dapat dilakukan dengan menggunakan *command diary*. Sebagai contoh jika ingin menyimpan keluaran dari perintah berikut

```
>> A = [ 1 2 3; 4 5 6; 7 8 9]

A =
     1     2     3
     4     5     6
     7     8     9
```

di direktori c:\backup dengan nama file data.txt maka dapat dilakukan dengan mengetik perintah berikut

```
diary 'D:\data.txt'
```

Jika ingin menutup menutup file data.txt maka dilakukan dengan mengetik perintah

```
diary off
```

Command window juga digunakan untuk memanggil *tool* Matlab seperti *editor*, *debugger* atau fungsi – fungsi yang lain. Ciri – ciri window ini adalah adanya prompt (>>) yang menyatakan Matlab siap menerima perintah. Perintah dapat berupa fungsi – fungsi pengaturan file maupun fungsi – fungsi *toolbox* Matlab sendiri. Berikut ini beberapa fungsi pengaturan file dalam Matlab diperlihatkan pada Tabel 2.1 berikut

Tabel 2.1 Fungsi - Fungsi Pengaturan File Pada Matlab

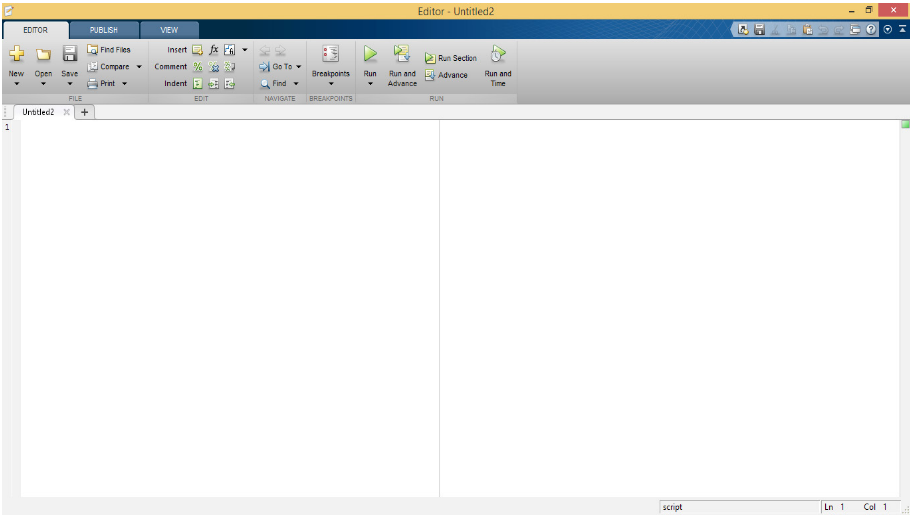
Fungsi	Keterangan
dir/ls	Perintah ini digunakan untuk melihat isi dari sebuah direktori yang aktif
cd	Perintah ini digunakan untuk melakukan perpindahan dari direktori aktif
pwd	Perintah ini digunakan untuk melihat direktori yang sedang aktif
mkdir	Perintah ini digunakan untuk membuat sebuah direktori
what	Perintah ini digunakan untuk melihat nama file yang berinisial m dalam direktori aktif
who	Perintah ini digunakan untuk melihat variabel yang sedang aktif
whos	Perintah ini digunakan untuk menampilkan nama setiap variabel
delete	Perintah ini digunakan untuk menghapus file
clear	Perintah ini digunakan untuk menghapus variabel
clc	Perintah ini digunakan untuk menghapus layar
demo	Perintah ini digunakan untuk mencoba beberapa tampilan demo yang disediakan oleh Matlab

b. Matlab Editor

Window ini merupakan tool yang disediakan oleh Matlab versi 5 ke atas yang berfungsi sebagai editor script Matlab (M- File). Walaupun sebenarnya script ini dalam pemograman Matlab dapat saja menggunakan editor lain seperti notepad, wordpad bahkan word. Untuk mengakses window M – file ini dapat dilakukan dengan cara mengetikan perintah berikut

```
>> edit
```

Dengan mengetikkan perintah *edit* tersebut maka akan diperoleh tampilan pada Gambar 2.2 berikut



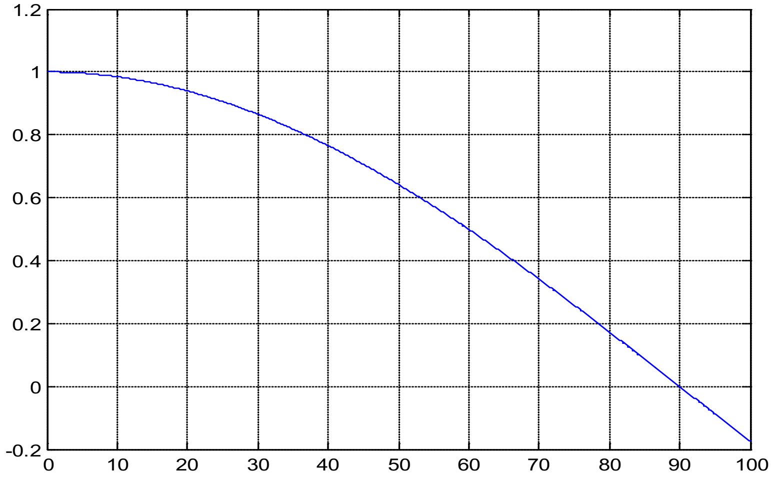
Gambar 2.2 Tampilan Matlab Editor

c. Figure Windows

Window ini adalah hasil visualisasi script Matlab Namun Matlab memberikan kemudahan bagi programmer untuk mengedit window ini sekaligus memberikan program khus untuk itu sehingga window ini selain berfungsi sebagai visualisasi keluaran dapat juga sekaligus menjadi media masukan yang interaktif. Script Matlab untuk visualisasi ini bisa ditulis pada *Matlab Command Window* dan *Matlab editor*. Untuk contoh visualisasi pada *Matlab Command Window* diperlihatkan dengan kode berikut

```
>> x = 0.00 : 0.10 : 100.00;
>> y = cos(x*pi/180);
>> plot(x,y)
>> grid on
```

Hasil yang diperoleh diperlihatkan pada Gambar 2.3 berikut



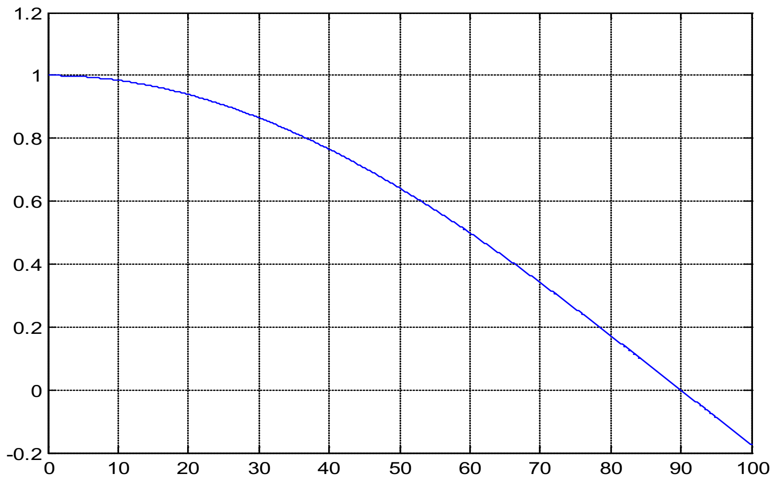
Gambar 2.3 Tampilan Grafik Cosinus

Untuk contoh visualisasi pada *Matlab Editor* diperlihatkan dengan kode berikut

```
clc
clear all
close all
close all hidden
%
x = 0.00 : 0.10 : 100.00;
y = cos(x*pi/180);
plot(x,y)
grid on
```

Hasil program

Hasil running program diperoleh grafik cosinus pada Gambar 2.4 berikut



Gambar 2.4 Tampilan Grafik Cosinus

d. Matlab Help Windows

Matlab menyediakan sistem *help* yang dapat diakses dengan perintah *help.* Misalkan untuk memperoleh informasi mengenai fungsi *fuzzy* maka pengguna hanya perlu mengetikkan perintah berikut

```
>> help fuzzy
```

dan dengan kemudian menekan enter maka di layar akan muncul informasi dalam bentuk teks pada layar Matlab berikut

```
Fuzzy Logic Toolbox
  Version 2.2.19 (R2014a) 27-Dec-2013

  GUI editors
    anfisedit   - ANFIS training and testing UI tool.
    findcluster - Clustering UI tool.
    fuzzy       - Basic FIS editor.
    mfedit      - Membership function editor.
    ruleedit    - Rule editor and parser.
    ruleview    - Rule viewer and fuzzy inference diagram.
    surfview    - Output surface viewer.

  Membership functions.
    dsigmf     - Difference of two sigmoid membership functions.
```

```
  gauss2mf   - Two-sided Gaussian curve membership function.
  gaussmf    - Gaussian curve membership function.
  gbellmf    - Generalized bell curve membership function.
  pimf       - Pi-shaped curve membership function.
  psigmf     - Product of two sigmoid membership functions.
  smf        - S-shaped curve membership function.
  sigmf      - Sigmoid curve membership function.
  trapmf     - Trapezoidal membership function.
  trimf      - Triangular membership function.
  zmf        - Z-shaped curve membership function.

Command line FIS functions
  addmf       - Add membership function to FIS
  addrule     - Add rule to FIS.
  addvar      - Add variable to FIS.
  defuzz      - Defuzzify membership function.
  evalfis     - Perform fuzzy inference calculation.
  evalmf      - Generic membership function evaluation.
  gensurf     - Generate FIS output surface.
  getfis      - Get fuzzy system properties.
  mf2mf       - Translate parameters between functions.
  newfis      - Create new FIS.
  parsrule    - Parse fuzzy rules.
  plotfis     - Display FIS input-output diagram.
  plotmf      - Display all membership functions for one variable.
  readfis     - Load FIS from disk.
  rmmf        - Remove membership function from FIS.
  rmvar       - Remove variable from FIS.
  setfis      - Set fuzzy system properties.
  showfis     - Display annotated FIS.
  showrule    - Display FIS rules.
  writefis    - Save FIS to file.

Advanced techniques
  anfis       - Training routine for Sugeno-type FIS (MEX only).
  fcm         - Find clusters with fuzzy c-means clustering.
  genfis1     - Generate FIS matrix using generic method.
  genfis2     - Generate FIS matrix using subtractive clustering.
  Subclust    - Estimate cluster centers with subtractive clustering.

Miscellaneous functions
 Convertfis   - Convert v1.0 fuzzy matrix to v2.0 fuzzy structure.
 discfis      - Discretize a fuzzy inference system.
 evalmmf      - For multiple membership functions evaluation.
 fstrvcat     - Concatenate matrices of varying size.
 fuzarith     - Fuzzy arithmatic function.
 findrow      - Find the rows of a matrix that match the input
```

```
string.
   genparam   - Generates initial premise parameters for ANFIS
                learning.
    probor    - Probabilistic OR.
    sugmax    - Maximum output range for a Sugeno system.

  GUI helper files
    cmfdlg    - Add customized membership function dialog.
    cmthdlg   - Add customized inference method dialog.
    fisgui    - Generic GUI handling for the Fuzzy Logic Toolbox
    gfmfdlg   - Generate fis using grid partition method dialog.
    mfdlg     - Add membership function dialog.
    mfdrag    - Drag membership functions using mouse.
    popundo   - Pull the last change off the undo stack.
    pushundo  - Push the current FIS data onto the undo stack.
    savedlg   - Save before closing dialog.
    statmsg   - Display messages in a status field.
    updtfis   - Update Fuzzy Logic Toolbox GUI tools.
    wsdlg     - Open from/save to workspace dialog.


fuzzy is both a directory and a function.

 fuzzy  Basic FIS editor.
    The FIS Editor displays high-level information about a
    Fuzzy Inference System. At the top is a diagram of the
    system with each input and output clearly labeled. By
    double-clicking on the input or output boxes, you can bring
    up the Membership Function Editor. Double-clicking on the
    fuzzy rule box in the center of the diagram will bring up
    the Rule Editor.

    Just below the diagram is a text field that displays the
    name of the current FIS. In the lower left of the window are
    a series of popup menus that allow you to specify the various
    functions used in the fuzzy implication process. In the lower
    right are fields that provide information about the current
    variable. The current variable is determined by clicking once
    on one of the input or output boxes.

    See also mfedit, ruleedit, ruleview, surfview, anfisedit.

    Reference page in Help browser
       doc fuzzy
```

2.4 Komentar dan Tanda Baca Matlab

Semua teks sesudah tanda % dianggap sebagai statemen komentar dengan contoh sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Plot Grafik Cosinus
x = 0.00 : 0.10 : 100.00;
y = cos(x*pi/180);
plot(x,y)
grid on
```

Statemen sesudah tanda % dianggap sebagai sebuah komentar. Statemen ini berguna untuk dokumentasi apa yang sudah dikerjakan. Tanda titik koma (;) dalam Matlab berguna untuk mencegah menampilkan hasil.

2.5 Matlab dan Matematika Sederhana

Ada tiga tipe bilangan di dalam Matlab yaitu bilangan bulat, bilangan real dan bilangan kompleks. Adapun contoh representasi masing – masing bilangan diperlihatkan pada contoh – contoh dibawah ini.

Untuk contoh bilangan bulat :

```
>> X = 10
X =
    10

>> X = 25

X =
    25
```

Untuk contoh bilangan real :

```
>> Y = 20.2500
Y =
   20.2500

>> Z = 15.2500
Z =
   15.2500
```

Untuk bilangan kompleks yang bentuk umumnya adala z = a + bi, salah satu kelebihan Matlab adalah tidak memerlukan penanganan khusus untuk bilangan kompleks dan untuk bilangan ini diberi tanda i atau j dengan contoh berikut

```
>> y = sqrt(-8)
y =
   0.0000 + 2.8284i

>> real(y)
ans =
     0

>> imag(y)
ans =
    2.8284

>> abs(y)
ans =
    2.8284

>> angle(y)
ans =
    1.5708
```

2.6 Fungsi Dasar Pada Matlab

Selain penambahan, pengurangan, perkalian, pembagian dan pemangkatan sering dibutuhkan rumus aritmatika yang lain. Sebagai contoh perhitungan yang membutuhkan fungsi trigonometri, logaritma dan fungsi analisis data juga disediakan dalam Matlab. Pada bagian ini akan dibahas fungsi dasar pada Matlab yang terdiri dari fungsi Matematika dasar, fungsi trigonometri dan fungsi analisis data.

2.6.1. Fungsi Matematika Dasar

Fungsi matematika dasar adalah fungsi yang digunakan untuk melakukan sejumlah perhitungan umum seperti yang diperlihatkan pada Tabel 2.2 berikut

Tabel 2.2 Fungsi - Fungsi Matematika Dasar Pada Matlab

Fungsi	Keterangan
abs	Fungsi ini berguna untuk menghitung nilai abosolut
ceil	Fungsi ini berguna untuk membulatkan bilangan ke bilangan bulat terdekat menu plus tak terhingga
exp	Fungsi ini berguna memperoleh nilai dari e pangkat bilangan tertentu
fix	Fungsi ini berguna untuk membulatkan bilangan ke bilangan bulat terdekat menuju nol
floor	Fungsi ini berguna untuk membulatkan bilangan ke bilangan bulat terdekat menuju minus tak berhingga
gcd	Fungsi ini berguna untuk menghitung nilai faktor pembagi terbesar
isprime	Fungsi ini berguna untuk menghasilkan true jika merupakan bilangan prima
log10	Fungsi ini berguna untuk menghitung logaritma suatu bilangan untuk dasar 10
mod	Fungsi ini berguna untuk menghitung nilai modus
primes	Fungsi ini berguna untuk menghasilkan daftar bilangan prima
rem	Fungsi ini berguna untuk menghitung nilai remainder
round	Fungsi ini berguna untuk membulatkan bilangan ke bilangan bulat terdekat
sqrt	Fungsi ini berguna untuk menghitung akar pangkat dua dari suatu bilangan

Adapun contoh – contoh untuk masing – masing fungsi sebagai berikut

```
>> abs(-10)
ans =
    10

>> ceil(5.89)
ans =
```

```
     6

>> ceil(5.23)
ans =
     6

>> ceil(-6.89)
ans =
    -6

>> exp(5)
ans =
  148.4132

>> A = [ -2.3000  -0.2400   3.4000   5.6000  7.0000 2.4000 +
j*3.6000]

A =
Columns 1 through 5
-2.3000 + 0.0000i  -0.2400 + 0.0000i   3.4000 + 0.0000i
5.6000 + 0.0000i   7.0000 + 0.0000i

Column 6
   2.4000 + 3.6000i

>> fix(A)
ans =
Columns 1 through 5
-2.0000 + 0.0000i   0.0000 + 0.0000i   3.0000 + 0.0000i
5.0000 + 0.0000i   7.0000 + 0.0000i

Column 6
2.0000 + 3.0000i

>> floor(A)
ans =
Columns 1 through 5
-3.0000 + 0.0000i  -1.0000 + 0.0000i   3.0000 + 0.0000i
5.0000 + 0.0000i   7.0000 + 0.0000i

Column 6
   2.0000 + 3.0000i

>> gcd(10,50)
ans =
    10
```

```
>> isprime(7)
ans =
     1

>> isprime(9)
ans =
     0

>> log(100)
ans =
    4.6052

>> log10(1000)
ans =
     3

>> mod(15,4)
ans =
     3

>> mod(-3,5)
ans =
     2

>> primes(15)
ans =
     2     3     5     7    11    13

>> rem(20,40)
ans =
     20

>> round(A)
ans =
Columns 1 through 5
-2.0000 + 0.0000i   0.0000 + 0.0000i   3.0000 + 0.0000i
6.0000 + 0.0000i   7.0000 + 0.0000i
Column 6
2.0000 + 4.0000i

>> sqrt(25)
ans =
     5
>> sqrt(-25)
ans =
   0.0000 + 5.0000i
```

2.6.2. Fungsi Trigonometri

Fungsi trigonometri banyak digunakan terkait dengan sudut. Nilai perhitungan fungsi trigonometri sudut dalam radian. Matlab menyediakan fungsi trigonometri seperti yang diperlihatkan pada Tabel 2.3 berikut

Tabel 2.3 Fungsi - Fungsi Trigonometri Pada Matlab

Fungsi	Keterangan
cos	Fungsi ini berguna untuk menghitung cosinus suatu bilangan dimana bilangan dinyatakan dalam satuan radian
sin	Fungsi ini berguna untuk menghitung sinus suatu bilangan dimana bilangan dinyatakan dalam satuan radian
tan	Fungsi ini berguna untuk menghitung tangen suatu bilangan dimana bilangan dinyatakan dalam satuan radian
acos	Fungsi ini berguna untuk menghitung arccosinus suatu bilangan dimana bilangan harus bernilai antara -1 sampai dengan 1 sedangkan hasil perhitungan berupa sudut dalam satuan radian
asin	Fungsi ini berguna untuk menghitung arcsinus suatu bilangan dimana bilangan harus bernilai antara -1 sampai dengan 1 sedangkan hasil perhitungan berupa sudut dalam satuan radian
atan	Fungsi ini berguna untuk menghitung arctangen suatu bilangan dan hasil perhitungan berupa sudut dalam satuan radian
cosh	Fungsi ini berguna untuk menghitung cosinus hiperbolik dari suatu sudut dalam satuan radian
sinh	Fungsi ini berguna untuk menghitung sinus hiperbolik dari suatu sudut dalam satuan radian
tanh	Fungsi ini berguna untuk menghitung tangen hiperbolik dari suatu sudut dalam satuan radian
cosd	Fungsi ini berguna untuk menghitung sinus suatu bilangan dimana bilangan dinyatakan dalam satuan derjat
sind	Fungsi ini berguna untuk menghitung sinus suatu bilangan dimana bilangan dinyatakan dalam satuan derjat
tand	Fungsi ini berguna untuk menghitung tangen suatu bilangan dimana bilangan dinyatakan dalam satuan derjat

sec	Fungsi ini berguna untuk menghitung 1/cos(x) suatu bilangan dimana bilangan dinyatakan dalam satuan radian
csc	Fungsi ini berguna untuk menghitung 1/sin(x) suatu bilangan dimana bilangan dinyatakan dalam satuan radian
cot	Fungsi ini berguna untuk menghitung 1/tan(x) suatu bilangan dimana dinyatak dalam satuan radian

Adapun contoh – contoh untuk masing – masing fungsi sebagai berikut

```
>> cos(120*(pi/180)) % Menghitung Cosinus (120)
ans =
   -0.5000

>> sin(120*(pi/180)) % Menghitung Sinus (120)
ans =
    0.8660

>> tan(120*(pi/180)) % Menghitung Tangen(120)
ans =
   -1.7321

>> acos(0.75)
ans =
    0.7227

>> asin(0.75)
ans =
    0.8481

>> atan(0.75)
ans =
    0.6435

>> cosh(120*(pi/180)) % Menghitung Cosinus Hiperbolik (120)
ans =
    4.1218

>> sinh(120*(pi/180)) % Menghitung Sinus Hiperbolik (120)
ans =
    3.9987

>> tanh(120*(pi/180)) % Menghitung Tangen Hiperbolik (120)
ans =
    0.9701
```

```
>> cosd(45)  % Menghitung cos(45) dalam satuan derjat
ans =
    0.7071

>> sind(45)  % Menghitung sin(45) dalam satuan derjat
ans =
    0.7071

>> tand(45)  % Menghitung tan(45) dalam satuan derjat
ans =
     1

>> csc(75*(pi/180))
ans =
    1.0353

>> sec(75*(pi/180))
ans =
    3.8637

>> cot(75*(pi/180))
ans =
    0.2679
```

2.6.3. Fungsi Analisis Data

Matlab menyediakan sejumlah fungsi penting untuk digunakan dalam menganalisa data. Adapun fungsi – fungsi untuk analisis data ini diperlihatkan pada Tabel 2.4 berikut

Tabel 2.4 Fungsi - Fungsi Analisis Data Pada Matlab

Fungsi	Keterangan
max	Fungsi ini berguna untuk menghasilkan nilai terbesar dari suatu vektor atau matriks
min	Fungsi ini berguna untuk menghasilkan nilai terkecil dari suatu vektor atau matriks
mean	Fungsi ini berguna untuk menghitung nilai rata –rata
median	Fungsi ini berguna untuk menghitung nilai tengah
std	Fungsi ini berguna untuk menghitung nilai standard deviasi
var	Fungsi ini berguna untuk menghitung nilai varian

corrcoef	Fungsi ini berguna untuk menghitung keofesien korelasi
sort	Fungsi ini berguna untuk mengurutkan data v
sum	Fungsi ini berguna untuk menghasilkan jumlah dari elemen suatu vektor atau menghasilkan sebuah vektor yang berisi jumlah setiap kolom dari suatu matrik
prod	Fungsi ini berguna untuk menghasilkan hasil perkalian elemen suatu vektor atau menghasilkan sebuah vektor yang berisi hasil perkalian setiap kolom dari suatu matrik

Adapun contoh – contoh untuk masing – masing fungsi sebagai berikut

```
>> x = [ 1 3 5 7 9]
x =
     1     3     5     7     9

>> max(x)  % Menentukan nilai maksimum
ans =
     9

>> min(x)  % Menentukan nilai minimum
ans =
     1

>> mean(x)
ans =
     5

>> median(x)
ans =
     5

>> std(x)
ans =
    3.1623

>> var(x)
ans =

    10

>> corrcoef(x)
ans =
     1
```

```
>> sort(x)
ans =
     1     3     5     7     9

>> sum(x)
ans =
    25

>> prod(x)
ans =
   945

>> y = [ 1 3 2; 4  5 7; 8  9 2]
y =
     1     3     2
     4     5     7
     8     9     2

>> max(y)
ans =
     8     9     7

>> min(y)
ans =
     1     3     2

>> mean(y)
ans =
    4.3333    5.6667    3.6667

>> median(y)
ans =
     4     5     2

>> std(y)
ans =
    3.5119    3.0551    2.8868

>> var(y)
ans =
   12.3333    9.3333    8.3333

>> corrcoef(y)
ans =
    1.0000    0.9942   -0.0822
    0.9942    1.0000   -0.1890
```

```
   -0.0822   -0.1890    1.0000

>> sort(y)
ans =
     1     3     2
     4     5     2
     8     9     7

>> sum(y)
ans =
    13    17    11

>> prod(y)
ans =
    32   135    28
```

2.7 Rangkuman

Matlab adalah sebuah program untuk analisis dan komputasi numerik serta merupakan suatu bahasa pemograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matrik. Selain itu Matlab banyak digunakan untuk matematika dan komputasi, pengembangan dan algoritma, pemograman pemodelan, simulasi dan pembuatan prototipe, analisa data, eksplorasi dan visualisasi, analisa numerik dan pengembangan aplikasi teknik. Pada bagian ini hanya dibahas tentang matlab dan matematika sederhana serta fungsi – fungsi dasar pada Matlab diantaranya fungsi matematika dasar, fungi trigonometri dan fungsi analisis data.

2.8 Soal - Soal

Soal 2.1: Dengan menggunakan Matlab, deskripsikan matrik dan vektor pada persamaan (2.1) s/d (2.6) berikut

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 2 & 4 & 3 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.1)$$

$$B = \begin{bmatrix} -5x & \log(2x) + 7\sin(3y) \\ 3i & 4 - 2i \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.2)$$

$$C = \begin{bmatrix} 1 & 5 & 3 & 4 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.3)$$

$$D = \begin{bmatrix} 1 & 2i & 3+5i & 4 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.4)$$

$$E = \begin{bmatrix} 1 \\ 3 \\ 2 \\ 4 \end{bmatrix} \quad \text{..........} \quad (2.5)$$

$$F = \begin{bmatrix} 1 \\ 2i \\ 3+5i \\ 4 \end{bmatrix} \quad \text{..........} \quad (2.6)$$

Soal 2.2: Dengan menggunakan Matlab, visualisasikan kode Matlab berikut

```
>> x = 0.00 : 0.50 : 100.00;
>> y = sin(x*pi/180);
>> plot(x,y)
>> grid on
```

Soal 2.3: Dengan menggunakan Matlab, hitung nilai – nilai dari persamaan (2.7) s/d (2.12) berikut

a. $|-50|$ .... (2.7)

b. e^{-2} .... (2.8)

c. e^{-2} .... (2.9)

d. $\ln(125)$ .... (2.10)

e. $\log(2000)$ .... (2.11)

f. $\sqrt{49}$ .... (2.12)

Soal 2.4: Dengan menggunakan Matlab, hitung nilai – nilai dari persamaan (2.13) s/d (2.24) berikut

a. $\cos(150^0)$ .... (2.13)

b. $\sin(150^0)$ .... (2.14)

c. $\tan\left(150^0\right)$ .... (2.15)

d. $\cos^{-1}\left(0.8660\right)$ .... (2.16)

e. $\sin^{-1}\left(0.8660\right)$ .... (2.17)

f. $\tan^{-1}\left(0.8660\right)$ .... (2.18)

g. $\cosh\left(150^0\right)$ .... (2.19)

h. $\sinh\left(150^0\right)$ .... (2.20)

i. $\tanh\left(150^0\right)$ .... (2.21)

j. $\mathrm{cosec}\left(150^0\right)$ .... (2.22)

i. $\sec\left(150^0\right)$ .... (2.23)

j. $\mathrm{cotangen}\left(150^0\right)$ .... (2.24)

Soal 2.5: Dengan menggunakan Matlab, untuk data – data pada persamaan (2.25) berikut

$x = [1.000 \quad 4.0000 \quad 3.0000 \quad 2.0000 \quad 5.0000 \quad 6.0000]$ .... (2.25)

Hitung

a. Nilai maksimum
b. Nilai minimum
c. Nilai mean
d. Nilai median
e. Nilai varian
f. Nilai koefesien korelasi

Soal 2.6: Dengan menggunakan Matlab, untuk matrik pada persamaan (2.26) berikut

$$y = \begin{bmatrix} 0.8147 & 0.9134 & 0.2785 \\ 0.9058 & 0.6324 & 0.5469 \\ 0.1270 & 0.0975 & 0.9575 \end{bmatrix} \quad \text{..........} \quad (2.26)$$

Hitung

a. Nilai maksimum
b. Nilai minimum
c. Nilai mean
d. Nilai median
e. Nilai varian
f. Nilai koefesien korelasi

BAB III
VARIABEL DAN TIPE DATA PADA MATLAB

3.1 Pendahuluan

Pada bagian ini dibahas tentang variabel dan tipe data pada Matlab. Materi yang dibahas meliputi variabel, string, skalar, array sel, struktur array dan array multidimensi serta pembahasan diakhiri dengan rangkuman dan soal – soal.

3.2 Variabel

Secara formal tidak dibutuhkan pendeklarasian suatu variabel baru dalam Matlab. Suatu variabel sederhana diciptakan dengan suatu *assignment*. Setiap variabel numerik dibuat baru maka selalu bertipe *Double*. Misalnya bilangan real diperkirakan dengan presisi tertinggi yang mungkin diperoleh dengan melakukan perubahan tipe bilangan menjadi *single*.

Nama variabel dimulai dengan huruf. Huruf yang digunakan bisa berupa huruf kapital maupun huruf kecil serta dapat diikuti oleh huruf, angka atau *underscore*. Selain itu nama variabel tidak boleh mengandung spasi. Adapun contoh penggunaan variabel diperlihatkan pada contoh – contoh berikut

```
>> Harga = 2500
Harga =
        2500
>> Harga_Satuan = 50000
Harga_Satuan =
       50000

>> x = 4
x =
     4

>> y = 2
y =
     2

>> y = y + 1
y =
     3
```

3.3 String

String dalam Matlab adalah tipe data yang terdiri dari huruf dan nilai – nilai ascii yang ditampilkan representasinya. String adalah teks yang diawali dan diakhiri dengan tanda apostrof. Contoh – contoh data dalam tipe string diperlihatkan pada Gambar.

```
>> S = 'Reri Afrianita'
S =
Reri Afrianita

>> size(S)
ans =
     1     14
```

Untuk melihat representasi ASCII karakter string dapat dilakukan dengan operasi aritmatika terhadap string atau melakukan konversi dengan fungsi double. Adapun contoh representasi ASCII karakter diperlihatkan pada contoh – contoh berikut

```
>> double(S)
ans =
82   101   114   105    32    65   102   114   105    97   110
105   116    97
```

atau dengan kode berikut

```
>> abs(S)
ans =
82   101   114   105    32    65   102   114   105    97   110
105   116    97
```

Selain itu Matlab menyediakan fungsi balikan dengan contoh berikut

```
>> char(S)
ans =
Reri Afrianita
```

String merupakan array numerik dengan atribut khusus, string dapat dimanipulasi menggunakan metoda manipulasi array yang tersedia dalam Matlab dengan contoh berikut

```
>> t = S(6:8)
t =
Afr
```

Jika kata tersebut dibalik maka kode Matlabnya sebagai berikut

```
>> t = S(8:-1:6)
t =
rfA
```

Jika menggunakan operator transposes maka `Afr` akan dibaca dalam format kolom yaitu

```
>> t = S(6:8)'
t =
A
f
r
```

Penggabungan String dapat dilakukan dengan mengikuti aturan penggabungan array. Adapun contoh penggabungan array sebagai berikut

```
>> a = 'Saya akan belajar dengan rajin'
a =
Saya akan belajar dengan rajin

>> b = 'dan Saya akan rajin menulis buku'
b =
dan Saya akan rajin menulis buku

>> c = [a b]
c =
Saya akan belajar dengan rajin dan Saya akan rajin menulis buku
```

Selain itu ada beberapa fungsi – fungsi string lain diantaranya

1. Fungsi *disp*

Fungsi disp memungkinkan untuk menampilkan string tanpa menampilkan nama variabelnya. Contoh penggunaan fungsi ini diperlihatkan dengan kode berikut

```
>> disp(a)
Saya akan belajar dengan rajin
>> disp(b)
dan Saya akan rajin menulis buku
```

Fungsi *disp* ini berguna untuk menampilkan teks bantuan dalam suatu file script seperti yang diperlihatkan pada contoh berikut

```
>> disp('Saya sedang menulis sebuah buku')
Saya sedang menulis sebuah buku
```

Syarat menggunakan fungsi *disp* ini adalah isi didalamnya harus merupakan strings. Jika ingin menampilkan sebuah angka maka terlebih dahulu harus diubah menjadi bentuk string dengan menggunakan fungsi *num2str()*. Adapun contoh penggunaan fungsi tersebut sebagai berikut

```
>> No_HP = 081277231272;
>> disp(['No. Handphone saya adalah ',num2str(No_HP)])
No. Handphone saya adalah 81277231272
```

2. Fungsi *input*

Beberapa contoh penggunaan fungsi input ini diperlihatkan dengan kode – kode berikut

```
>> no_nim = input('Masukkan No nim ada !','s')
Masukkan No nim ada ! 95171042
no_nim =
95171042
>> disp('No. Nim yang anda masukan =');
No. Nim yang anda masukan =
>> disp(no_nim)
 95171042
```

3. Fungsi *fprintf*

Fungsi ini berfungsi untuk menampilkan keluaran di layar atau menyimpan keluaran di suatu external file. Adapun contoh penggunaan fungsi *fprintf* ini diperlihatkan pada contoh – contoh berikut

```
>> x = 'Hendri Sambodo';
>> y = 'Jl. Gajah Mada No. 40 Padang';
>> fprintf('%s\n',x)
Hendri Sambodo
>> fprintf('%s\n',x,y)
Hendri Sambodo
Jl. Gajah Mada No. 40 Padang
```

Contoh yang lain sebagai berikut

```
>> x = 2007.46567;
>> y = 2.1545;
>> k = 17;
>> fprintf('x = %8.3f  y = %8.3f  k = %2.0f',x,y,k)
x = 2007.466  y =    2.155  k = 17>>
```

3.4 Skalar

Skalar adalah nama lain dari data numerik. Dalam Matlab data skalar dapat dimanipulasi dengan menggunakan beberapa fungsi tipe string sebelumnya. Beberapa contoh dari skalar ini sebagai berikut

```
>> x = 0:0.01:2;
>> y = [x; cos(x)];
>> fid = fopen('D:Data_cos.txt','w');
>> fprintf(fid,'&6.3f %12.8f\n',y);
>> fclose(fid)
```

Matlab akan menyimpan tabel berisikan x dan cos(x) di file Data_cos.txt

3.5 Array Sel

Array sel merupakan suatu array yang bisa memuat "benda- benda" yang berbeda. "Benda" tersebut bisa saja skalar, vektor, matrik, string, struktur atau array sel yang lain. Jika diciptakan suatu matrik string, maka akan disibukkan dengan bagaimana dengan menyisipkan spasi agar setiap baris menjadi sama panjang. Dengan menggunakan suatu array sel maka dengan bebas digunakan untuk representasi yang diinginkan. Adapun contoh – contoh penggunaan array sel diperlihatkan pada contoh berikut

```
>> t = { 'Indonesia Tanah Airku';
'Danau Singkarak Terletak di Kabupaten Solok';
'Kota Bukittinggi Kota Wisata';
'Batusangkar Kota Budaya'}

t =
'Indonesia Tanah Airku'
'Danau Singkarak Terletak di Kabupaten Solok'
'Kota Bukittinggi Kota Wisata'
'Batusangkar Kota Budaya'
```

Sepasang tanda kurung kurawal menandakan suatu sel. Sel yang baru saja diciptakan adalah suatu array sel yang berdimensi 4 x 1.

```
>> whos
  Name       Size             Bytes  Class    Attributes
  t          4x1                678  cell

>> t(1)
ans =
    'Indonesia Tanah Airku'
```

```
>> t(2)
ans =
    'Danau Singkarak Terletak di Kabupaten Solok'

>> t(3)
ans =
    'Kota Bukittinggi Kota Wisata'

>> t(4)
ans =
    'Batusangkar Kota Budaya'
```

Sekarang satu elemen akan ditambahkan ke array sel tersebut dengan menempatkan suatu matrik yang berukuran 3 x 3 pada baris pertama dan kolom. Adapun kodenya diperlihatkan sebagai berikut

```
>> t{1,2} = spiral(3)
t =
    'Indonesia Tanah Airku'                          [3x3 double]
    'Danau Singkarak Terletak di Kabupaten...'                 []
    'Kota Bukittinggi Kota Wisata'                             []
    'Batusangkar Kota Budaya'                                  []
```

Matlab secara otomatis mengisi sisa sel pada kolom kedua dengan sel – sel kosong. Digunakan kurung kurawal t{1,2} untuk merujuk sel yang bersangkutan. Jika digunakan kurung biasa maka akan terjadi *error*.

Untuk mengekstrak kata `sangkar`, maka perlu dilakukan akses sel{4,1} , kemudian diperoleh karakter ke-5 sampai ke-11 dari isi sel tersebut. Adapun kodenya diperlihatkan sebagai berikut

```
>> t = { 'Indonesia Tanah Airku';
'Danau Singkarak Terletak di Kabupaten Solok';
'Kota Bukittinggi Kota Wisata';
'Batusangkar Kota Budaya'}

>> t{4,1}(5:11)
ans =
sangkar
```

3.6 Struktur Array

Struktur adalah array yang nama – namanya dipisahkan dengan tanda titik. Struktur bisa digunakan untuk menyimpan informasi – informasi yang berbeda jenis tetapi diikat dalam suatu struktur hirarki. Adapun contoh struktur diperlihatkan dengan kode berikut

```
clc
clear all
close all
close all hidden
%
Dosen.Nama = 'Heru Dibyo Laksono';
Dosen.NIP = '197701072005011002';
Dosen.Prodi = 'Teknik Elektro';
Dosen(2).Nama = 'Reri Afrianita';
Dosen(2).NIP = '197704172006011001';
Dosen(2).Prodi = 'Teknik Lingkungan';
%
D1 = Dosen(1)
D2 = Dosen(2)
```

Hasil program

```
D1 =
     Nama: 'Heru Dibyo Laksono'
      NIP: '197701072005011002'
    Prodi: 'Teknik Elektro'
D2 =
     Nama: 'Reri Afrianita'
      NIP: '197704172006011001'
    Prodi: 'Teknik Lingkungan'
```

Struktur `Dosen` memiliki tiga bidang Nama, NIP dan Prodi.

Selain itu akan dibuat suatu struktur menggunakan fungsi *`struct`*. Dengan menggunakan fungsi *`struct`* ini akan dibuat suatu database pengamatan meteorolgi dengan menggunakan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
meteor = struct('Situs',{'Rinjani','Merapi'},...
    'Waktu',{2.34},...
    'Temperatur',{24  19},...
```

```
    'Tekanan',{1023  1015})
%
m1 = meteor(1)
%
m2 = meteor(2)
```

Hasil dari kode Matlab diperoleh tampilan sebagai berikut

```
meteor =
1x2 struct array with fields:
    Situs
    Waktu
    Temperatur
    Tekanan
m1 =
         Situs: 'Rinjani'
         Waktu: 2.3400
    Temperatur: 24
       Tekanan: 1023
m2 =
         Situs: 'Merapi'
         Waktu: 2.3400
    Temperatur: 19
       Tekanan: 1015
```

Seandainya ada data terbaru pada situs Merapi yang diambil pada pukul 08.00 s/d 11.00, bisa dilakukan penambahan dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
meteor = struct('Situs',{'Rinjani','Merapi'},...
    'Waktu',{2.34},...
    'Temperatur',{24  19},...
    'Tekanan',{1023  1015})
%
meteor(2).Waktu(2:3) = [ 8  11];
meteor(2).Temperatur(2:3) = [16.5000  15.3000];
%
T = meteor(2).Temperatur
```

Hasil dari kode Matlab diperoleh tampilan sebagai berikut

```
meteor =
1x2 struct array with fields:
    Situs
    Waktu
```

```
    Temperatur
    Tekanan
T =
   19.0000   16.5000   15.3000
```

3.7 Array Multidimensi

Suatu matrik 3 x 3 x 3 dapat dihasilkan dengan kode sebagai berikut

```
>> A = [ 1  2  3; 4  5  6; 7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
>> A(:,:,2) = A*2
A(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
A(:,:,2) =
     2     4     6
     8    10    12
    14    16    18
>> A(:,:,3) = eye(3)
A(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
A(:,:,2) =
     2     4     6
     8    10    12
    14    16    18
A(:,:,3) =
     1     0     0
     0     1     0
     0     0     1
```

Array multidimensi harus memiliki panjang kolom yang sama. Jika ditugaskan suatu nilai tunggal kepada suatu matrik maka MatlaB akan secara otomatis mengekspansi seperti yang diharapkan.

```
>> A(:,:,3) = 3
A(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
A(:,:,2) =
```

```
     2     4     6
     8    10    12
    14    16    18
A(:,:,3) =
     3     3     3
     3     3     3
     3     3     3
```

Pemberian nomor indeks array multidimensi sama seperti pada array dua dimensi. Adapun contoh pemberian indeks ini diperlihatkan dengan kode Matlab berikut

```
>> A(2,:,1)
ans =
     4     5     6

>> A(2,:,2)
ans =
     8    10    12

>> A(2,:,:)
ans(:,:,1) =
     4     5     6
ans(:,:,2) =
     8    10    12
ans(:,:,3) =
     3     3     3
```

Data dapat dihilangkan dari array multidimensi menggunakan matrik kosong dengan kode sebagai berikut

```
>> A(:,:,2) = []
A(:,:,1) =

     1     2     3
     4     5     6
     7     8     9

A(:,:,2) =
     3     3     3
     3     3     3
     3     3     3
```

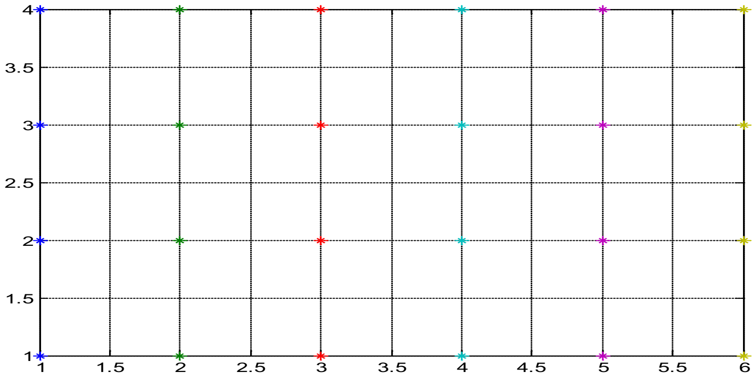
Elemen – elemen dapat secara kolom diekstrak dari suatu array multidimensi dengan cara yang sama seperti pada array dua dimensi. Adapun contoh tersebut diperlihatkan dengan kode berikut

```
>> A(:)'
ans =
Columns 1 through 15

1     4      7      2      5      8      3      6      9      3      3
3      3     3      3

Columns 16 through 18
3      3      3
```

Selain itu dengan fungsi – fungsi Matlab dapat dihasilkan grid – grid dengan titik –titik yang mempuyai spasi sama. Adapun kode Matlab diperlihatkan pada contoh berikut

```
>> [x,y] = meshgrid(1:6,1:4)
x =
     1      2      3      4      5      6
     1      2      3      4      5      6
     1      2      3      4      5      6
     1      2      3      4      5      6


y =
     1      1      1      1      1      1
     2      2      2      2      2      2
     3      3      3      3      3      3
     4      4      4      4      4      4
>> plot(x,y,'*')
>> grid on
```

Grafik yang diperoleh diperlihatkan pada Gambar 3.1 berikut

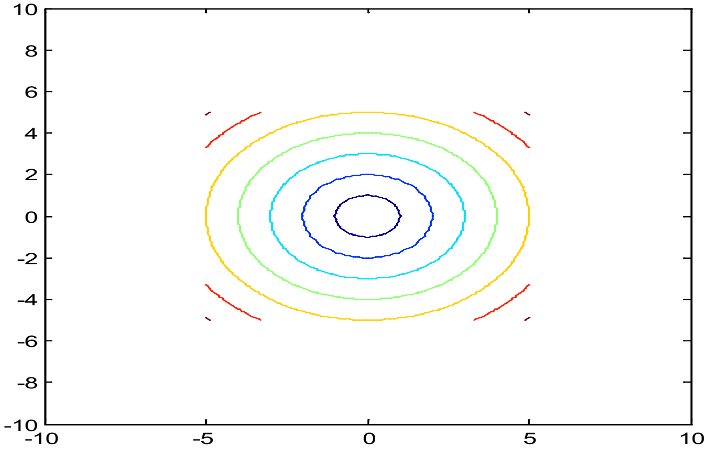


Gambar 3.1 Grafik Fungsi Meshgrid

Matrik seperti itu dapat dipakai sebagai variabel – variabel di dalam fungsi x dan fungsi y dengan kode sebagai berikut

```
>> clf
>> plot(x,y,'o')
>> axis([0.90  5.00  0.90   3.00])
>> [x,y] = meshgrid(linspace(-5,5),linspace(-5,5));
>> r = sqrt(x.^2 + y.^2);
>> contour(x,y,r)
>> axis equal
>> axis([-10.00  10.00  -10.00   10.00])
```

Hasil program



Gambar 3.2 Grafik Fungsi Contour

3.8 Rangkuman

Beberapa tipe data yang digunakan pada Matlab diantaranya variabel, string, skalar, array sel, struktur array dan array mutidimensi. Variabel adalah tempat dimana dapat dilakukan pengisian atau pengosongan nilai dan memanggil kembali apabila dibutuhkan. Setiap variabel akan mempunyai nama dan nilai. String merupakan array dari sekumpulan karakter yang berukuran 1 x n. Dalam matlab, string diibaratkan sebuah array dari kumpulan karakter. Setiap karakter direpresentasikan sebagai sebuah nilai ASCII. Skalar merupakan matrik yang berisi satu elemen. Suatu array adalah sebuah struktur data yang terdiri atas banyak variabel dengan tipe data sama, dimana masing-masing elemen variabel mempunyai nilai indeks. Setiap elemen array mampu untuk menyimpan satu jenis data dimana array multidimensi ini terdiri atas baris dan kolom. Indek pertama adalah baris dan yang keduan adalah kolom.

3.9 Soal - Soal

Soal 3.1: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> Panjang = 5;
>> Lebar = 20;
>> Luas = Panjang * Lebar
Luas =
   100
```

Soal 3.2: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> P = 'Thanisa Nashwa Azura'
P =
Thanisa Nashwa Azura
>> double(P)
ans =
Columns 1 through 17
84   104    97   110   105   115    97    32    78    97   115
104  119    97    32    65   122
Columns 18 through 20
117   114    97

>> abs(P)
ans =
Columns 1 through 17
84   104    97   110   105   115    97    32    78    97   115
104   119    97    32    65   122
Columns 18 through 20
117   114    97
>> char(P)
ans =
Thanisa Nashwa Azura
>> t = P(5:8)
t =
isa
```

Soal 3.3: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> a = 'Heru Dibyo Laksono'
```

```
a =
Heru Dibyo Laksono
>> b = 'Jurusan Teknik Elektro'
b =
Jurusan Teknik Elektro
>> c = 'Fakultas Teknik'
c =
Fakultas Teknik
>> d = 'Universitas Andalas'
d =
Universitas Andalas
```

Soal 3.4: Dengan menggunakan Matlab, lanjutkan deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> disp(a)
Heru Dibyo Laksono
>> disp(b)
Jurusan Teknik Elektro
>> disp(c)
Fakultas Teknik
>> disp(d)
Universitas Andalas
```

Soal 3.5: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> disp('Reri Afrianita')
Reri Afrianita
>> disp('Jurusan Teknik Lingkungan')
Jurusan Teknik Lingkungan
>> disp('Fakultas Teknik')
Fakultas Teknik
>> disp('Universitas Andalas')
Universitas Andalas
```

Soal 3.6: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> No_Pendaftaran = 123450654321;
>> disp(['No. Pendaftaran saya adalah ',num2str(No_
Pendaftaran)])
No. Pendaftaran saya adalah 123450654321
```

Soal 3.7: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> No_nim = input(' Masukkan No Nim Anda : ','s')
Masukkan No Nim Anda : 95171040
No_nim =
95171040
```

Soal 3.8: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> x = 'Budi Sampurno';
>> y = 'PT. Sampoerna Indonesia';
>> fprintf('%s\n',x)
Budi Sampurno
>> fprintf('%s\n',y)
PT. Sampoerna Indonesia
```

Soal 3.9: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Command Window*

```
>> p = {'Objek Wisata di Sumatera Barat';
'Bukittingi dengan Objek Wisata Jam Gadang';
'Solok dengan Objek Wisata Danau Singkarak';
'Tanahdatar dengan Objek Wisata Istano Pagaruyung';
'Sawahlunto dengan Objek Wisata Tambang Dalam'}

p =
    'Objek Wisata di Sumatera Barat'
    'Bukittingi dengan Objek Wisata Jam Gadang'
    'Solok dengan Objek Wisata Danau Singkarak'
    'Tanahdatar dengan Objek Wisata Istano Pagaruyung'
    'Sawahlunto dengan Objek Wisata Tambang Dalam'
```

Soal 3.10: Dengan menggunakan Matlab, lanjutkan deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> whos
  Name      Size           Bytes  Class    Attributes
  p         5x1              968  cell

>> p(1)
ans =
    'Objek Wisata di Sumatera Barat'

>> p(2)
ans =
    'Bukittingi dengan Objek Wisata Jam Gadang'

>> p(3)
```

```
ans =
    'Solok dengan Objek Wisata Danau Singkarak'

>> p(4)
ans =
    'Tanahdatar dengan Objek Wisata Istano Pagaruyung'

>> p(5)
ans =
    'Sawahlunto dengan Objek Wisata Tambang Dalam'
```

Soal 3.11: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
Mahasiswa.Nama = 'Rudi Sujarwo';
Mahasiwa.NIM = '95171041';
Mahasiwa.Prodi = 'Teknik Mesin';
Mahasiswa(2).Nama = 'Rudi Sujarwi';
Mahasiwa(2).NIM = '95172042';
Mahasiwa(2).Prodi = 'Teknik Sipil';
Mahasiswa(3).Nama = 'Rudi Sukarwo';
Mahasiwa(3).NIM = '95173043';
Mahasiwa(3).Prodi = 'Teknik Industri';
Mahasiswa(4).Nama = 'Rudi Sukarwijo';
Mahasiwa(4).NIM = '95175044';
Mahasiwa(4).Prodi = 'Teknik Lingkungan';
Mahasiswa(5).Nama = 'Rudi Sukarmanto';
Mahasiwa(5).NIM = '95175045';
Mahasiwa(5).Prodi = 'Teknik Elektro';
%
M1 = Mahasiswa(1)
M2 = Mahasiswa(2)
M3 = Mahasiswa(3)
M4 = Mahasiswa(4)
M5 = Mahasiswa(5)
```

Soal 3.12: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
```

```
close all hidden
%
gunung = struct('Situs',{'Rinjani','Merapi'},...
     'Waktu',{2.34},...
     'Temperatur',{24  19},...
     'Tekanan',{1023  1015},...
     'Ketinggian',{2024 2012})
%
G1 = gunung(1)
G2 = gunung(2)
```

Soal 3.13: Dengan menggunakan Matlab, deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> A = [ 7  8  9; 4  2  6; 1 2 3]
A =
     7     8     9
     4     2     6
     1     2     3

>> A(:,:,2) = A*4
A(:,:,1) =
     7     8     9
     4     2     6
     1     2     3
A(:,:,2) =
    28    32    36
    16     8    24
     4     8    12

>> A(:,:,3) = eye(3)
A(:,:,1) =
     7     8     9
     4     2     6
     1     2     3
A(:,:,2) =
    28    32    36
    16     8    24
     4     8    12
A(:,:,3) =
     1     0     0
     0     1     0
     0     0     1
```

Soal 3.14: Dengan menggunakan Matlab, lanjutkan deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> A(:,:,3) = 4
A(:,:,1) =
     7     8     9
     4     2     6
     1     2     3


A(:,:,2) =
    28    32    36
    16     8    24
     4     8    12

A(:,:,3) =
     4     4     4
     4     4     4
     4     4     4
```

Soal 3.15: Dengan menggunakan Matlab, lanjutkan deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> A(2,:,1)
ans =
     4     2     6

>> A(2,:,2)
ans =
    16     8    24

>> A(2,:,:)
ans(:,:,1) =
     4     2     6
ans(:,:,2) =
    16     8    24
ans(:,:,3) =
     4     4     4
```

Soal 3.16: Dengan menggunakan Matlab, lanjutkan deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> A(:,:,2) = []
A(:,:,1) =

     7     8     9
     4     2     6
```

```
     1     2     3
A(:,:,2) =
     4     4     4
     4     4     4
     4     4     4
```

Soal 3.17: Dengan menggunakan Matlab, lanjutkan deskripsi kode – kode Matlab berikut pada *Matlab Command Window*

```
>> A(:)'
ans =
Columns 1 through 17
7     4     1     8     2     2     9     6     3     4     4
4     4     4     4     4     4
Column 18
4
```

Soal 3.18: Dengan menggunakan Matlab, deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clf
[x,y] = meshgrid(1:4,1:4)
plot(x,y,'o')
axis([0.90  5.00  0.90   3.00])
[x,y] = meshgrid(linspace(-5,5),linspace(-5,5));
r = sqrt(x.^3 + y.^3);
contour(x,y,r)
axis equal
axis([-10.00  10.00  -10.00   10.00])
grid on
```

BAB IV
VEKTOR DAN MATRIK PADA MATLAB

4.1 Pendahuluan

Bagian ini membahas tentang vektor dan matrik pada Matlab. Untuk vektor pembahasan meliputi notasi titik dua, ekstraksi bagian suatu vektor, vektor kolom dan transportasi, perkalian, pembagian dan pangkat terhadap vektor. Untuk matrik pembahasan meliputi matrik khusus, membuat matrik dan mengekstrak bagian matrik, operasi dan fungsi pada matriks. Pembahasan pada bab ini akan ditutup dengan rangkuman dan soal – soal.

4.2 Vektor

Baris vektor adalah daftar angka – angka yang dipisahkan oleh koma atau spasi. Vektor adalah contoh sederhana dari array. Elemen pertama mempuyai indeks 1. Jumlah masukan dikenal panjang vektor. Adapun contoh dari vektor dan menghitung panjang vektor diperlihatkan dengan kode sebagai berikut

```
>> p = [ 1 4 3 2 ]
p =
     1     4     3     2

>> length(p)
ans =
     4
```

Sejumlah operasi dapat dilakukan pada vektor. Suatu vektor dapat dikalikan dengan suatu skalar atau ditambahkan/dikurangkan ke/dari vektor lain yang sama panjang atau suatu bilangan dapat ditambahkan/dikurangkan ke atau dari suatu vektor. Seluruh operasi ini akan diproses elemen per elemen. Vektor dapat juga dibangun dari vektor yang telah ada sebelumnya. Adapun contoh – contoh dari operasi vektor diperlihatkan pada kode – kode berikut.

```
>> p = [ 1 5 3 2 ]
p =
     1     5     3     2

>> q = [ 3 5 2 7]
q =
     3     5     2     7

>> z = p + q
z =
     4    10     5     9
```

```
>> z1 = p + 2
z1 =
     3     7     5     4
```

4.2.1 Notasi Titik Dua dan Ekstraksi Bagian Suatu Vektor

Notasi titik dua (:) merupakan notasi penting untuk digunakan dalam menghasilkan vektor baris . Bentuk umum penulisan titik dua sebagai berikut

```
Awal : langkah : akhir
```

dimana

- Awal adalah nilai elemen pertama vektor
- Langkah adalah nilai pertambahan elemen, jika tidak disertakan otomatis pertambahan 1.
- Akhir adalah nilai akhir vektor.

Beberapa contoh aplikasi vektor dengan menggunakan Matlab diperlihatkan dengan kode berikut

```
>> 2 : 10
ans =
     2     3     4     5     6     7     8     9    10

>> 2 : 2 : 10
ans =
     2     4     6     8    10

>> -5 : 5
ans =
    -5    -4    -3    -2    -1     0     1     2     3     4
5

>> 1 : 0.5 : 5
ans =
1.0000    1.5000    2.0000    2.5000    3.0000    3.5000
4.0000    4.5000    5.0000

>> -5 : 1 : 10
ans =
-5    -4    -3    -2    -1     0     1     2     3     4     5
6      7     8     9    10
```

Bagian vektor dapat diekstrak dengan menggunakan notasi titik dua (:) dengan contoh berikut

```
>> x = [1:1:5, 7, 8, 9]

x =

     1     2     3     4     5     7     8     9
```

Untuk mengambil elemen vektor x pada posisi 3 sampai 6 dengan kode berikut

```
>> x(3:6)

ans =

     3     4     5     7
```

Untuk mengambil elemen vektor x pada posisi 2, 4 dan 6 dengan kode berikut

```
>> x(2:2:6)
ans =
     2     4     7
```

Untuk mengambil elemen vektor x pada posisi 4, 3, 2 dan 1 dengan kode berikut

```
>> x(4:-1:1)
ans =
     4     3     2     1
```

4.2.2 Vektor Kolom dan Transportasi

Untuk membuat vektor kolom dilakukan dengan memisahkan masukan dengan baris baru atau dengan tanda titik koma (;). Adapun contoh – contoh vektor kolom dilakukan dengan kode berikut

```
>> y = [3
        4
        5]

y =
     3
     4
     5
```

Dengan cara yang lain dengan menggunakan kode berikut

```
>> y = [3; 4; 5]
y =
     3
     4
     5
```

Operasi yang sama seperti vektor baris dapat dilakukan pada vektor kolom. Namun untuk menambahkan suatu vektor kolom secara langsung terhadap suatu vektor baris tidak bisa dilakukan. Agar penambahan suatu vektor kolom terhadap vektor baris bisa dilakukan maka harus dilakukan dulu transposisi. Tranposisi ini merupakan proses perubahan vektor kolom menjadi vektor baris atau sebaliknya. Hal ini diperlihatkan pada contoh – contoh dengan kode berikut

```
>> p = [4; 5; 6]
p =
     4
     5
     6

>> q = [ 1 2 3]
q =
     1     2     3

>> p + p
ans =
     8
    10
    12

>> p + q'
ans =
     5
     7
     9
```

4.2.3 Perkalian, Pembagian dan Pangkat Terhadap Vektor

Perkalian dua vektor bisa dilakukan jika kedua vektor tersebut mempuyai panjang yang sama. Adapun contoh perkalian vektor diperlihatkan dengan kode Matlab berikut

```
>> x = [ 1; 4; 8]
x =
```

```
     1
     4
     8

>> y = [ 2; 6; 9]
y =
     2
     6
     9

>> p = x'*y
p =
    98
```

Cara lain untuk menghitung perkalian elemen vektor menggunakan dot product (.*). Misalkan untuk dua vektor x dan vektor y yang sama panjang dilakukan perkalian elemen vektor dengan kode sebagai berikut

```
>> x = [ 4; 6; 8]
x =
     4
     6
     8

>> y = [ 2; 6; 8]
y =
     2
     6
     8

>> x.*y
ans =
     8
    36
    64

>> sum(x.*y)
ans =
   108
```

Operasi pangkat pada vektor diperlihatkan dengan kode sebagai berikut

```
>> x = [1 : 3 :15]

x =
     1     4     7    10    13
```

```
>> y = sqrt(x)
y =
    1.0000    2.0000    2.6458    3.1623    3.6056

>> y = cos(x)
y =

    0.5403   -0.6536    0.7539   -0.8391    0.9074

>> y = x.^2
y =
     1    16    49   100   169
```

Secara matematika tidak ada keterangan pembagian satu vektor dengan vektor yang lain, namun dalam Matlab dapat dilakukan dengan operator ./ untuk membagikan elemen – elemennya dengan catatan vektor harus berukuran sama dan bertipe sama. Adapun contoh pembagian vektor ini diperlihatkan dengan kode berikut

```
>> x = 1 : 1 : 5
x =
     1     2     3     4     5

>> y = 6 : 1 : 10
y =
     6     7     8     9    10

>> x./y
ans =
    0.1667    0.2857    0.3750    0.4444    0.5000
```

Selain itu operator ./ dapat juga digunakan untuk membagi suatu skalar dengan suatu dengan kode sebagai berikut

```
>> x = 1 : 10
x =
     1     2     3     4     5     6     7     8     9    10

>> 5 ./x
ans =
5.0000    2.5000    1.6667    1.2500    1.0000    0.8333    0.7143
0.6250    0.5556    0.5000
```

4.3 Matrik

Matrik adalah set bilangan nyata atau bilangan kompleks yang disusun dalam baris dan kolom sehingga membentuk jajaran persegi panjang. Suatu matrik yang memiliki m baris dan n kolom disebut matriks $m \times n$ dan disebut juga sebagai matrik yang memiliki orde $m \times n$. Perintah Matlab berikut untuk representasi matrik yang berorde 3×3 dengan elemen-elemen bernilai real dan matrik yang berorde 2×2 dengan elemen-elemen bernilai kompleks pada persamaan (4.1) dan (4.2) berikut

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 2 & 4 & 3 \end{bmatrix} \quad \text{...} \quad (4.1)$$

Matrik pada persamaan (4.1) dinyatakan dengan kode Matlab berikut

```
>> A = [ 1  2   3; 4  5  6; 2  4  3]
A =

     1     2     3
     4     5     6
     2     4     3
```

atau

```
>> A = [ 1,  2,   3,; 4,  5,  6; 2,  4,  3]

A =
     1     2     3
     4     5     6
     2     4     3
```

atau

```
>> A = [ 1  2  3
         4  5  6
         2  4  3]

A =
     1     2     3
     4     5     6
     2     4     3
```

4.3.1 Matrik Khusus

Ada sejumlah fungsi – fungsi khusus matrik yang terdapat pada Matlab. Fungsi – fungsi tersebut mempuyai penggunaan khusus diantaranya

- Matrik Kosong

Untuk matrik kosong ini diperlihatkan dengan kode berikut

```
>> E = [ ]
E =
     []

>> size(E)
ans =
     0     0
```

- Matrik Identitas

Untuk matrik identitas diperlihatkan dengan kode berikut

```
>> I = eye(5)
I =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
```

- Matrik Diagonal

Untuk matrik diagonal diperlihatkan dengan kode berikut

```
>> r = [ 1 2 3]
r =
     1     2     3
>> R = diag(r)
R =
     1     0     0
     0     2     0
     0     0     3
```

- Ekstrak Diagonal Matrik

Untuk ekstrak diagonal suatu matrik diperlihatkan dengan kode berikut

```
>> A = [ 4 5 7; 8 9 10; 3 2 1]
A =
     4     5     7
```

```
     8      9     10
     3      2      1

>> B = diag(A)
B =
     4
     9
     1
```

- Membuat Matrik dengan Elemen 1

Untuk membuat matrik dengan elemen – elemen bernilai satu diperlihatkan dengan kode berikut

```
>> B = ones(4,2)
B =
     1      1
     1      1
     1      1
     1      1
```

- Membuat Matrik dengan Elemen 0

Untuk membuat matrik dengan elemen – elemen bernilai nol diperlihatkan dengan kode berikut

```
>> C = zeros(3,4)
C =
     0      0      0      0
     0      0      0      0
     0      0      0      0
```

- Membuat Matrik dengan Elemen Random

Untuk membuat matrik dengan elemen – elemen bernilai random diperlihatkan dengan kode berikut

```
>> C = rand(4,5)
C =
    0.8147    0.6324    0.9575    0.9572    0.4218
    0.9058    0.0975    0.9649    0.4854    0.9157
    0.1270    0.2785    0.1576    0.8003    0.7922
    0.9134    0.5469    0.9706    0.1419    0.9595
```

4.3.2 Membuat Matrik dan Mengekstrak Bagian Matrik

Membuat matrik dan mengekstrak bagian matrik sering dibutuhkan untuk membuat matrik yang lebih besar dari beberapa matrik yang berukuran kecil. Adapun contoh – contoh untuk membuat matrik dan mengekstrak bagian matrik sebagai berikut

- Memasukkan Elemen Matrik A

```
>> A = [ 1; 3; 4]

A =
     1
     3
     4
```

- Memasukkan Elemen Matrik B

```
>> B = [ 4 5 6]

B =
     4     5     6
```

- Menggabungkan Matrik A dan Matrik B transpose

```
>> C = [A B']
C =
     1     4
     3     5
     4     6

>> size(C)
ans =
     3     2
```

- Memasukkan Elemen Matrik P

```
>> p = [ -1  2   3; 4   2  3]
p =
    -1     2     3
     4     2     3

>> size(p)
ans =
     2     3
```

- Memasukkan Elemen Matrik Q

```
>> Q = 1 : 3
Q =
     1     2     3

>> size(Q)
ans =
      1      3
```

- Memasukkan Elemen Matrik Q

```
>> q = 1 : 3
q =
     1     2     3

>> size(q)
ans =
      1      3
```

- Menggabung Matrik A dan Matrik Q serta menyimpan pada R

```
>> R = [p;q]
R =
    -1     2     3
     4     2     3
     1     2     3

>> size(R)
ans =
      2      3
```

- Memasukkan Elemen Matrik z

```
>>  z = [1  7; 2   6; 1   4]
z =
     1     7
     2     6
     1     4

>> size(z)
ans =
      3      2
```

- Menggabung Matrik R dan Matrik z

```
>> S = [R z]
S =
```

```
    -1     2     3     1     7
     4     2     3     2     6
     1     2     3     1     4

>> size(S)
ans =
      4      5
```

Bagian yang diekstrak dari suatu matrik memiliki cara yang sama dengan yang dilakukan terhadap vektor. Setiap elemen matrik diurutkan berdasarkan baris dan kolom sesuai kebutuhan. Secara matematika, elemen baris ke I dan kolom ke j dari suatu matrik A dituliskan dalam notasi Matlab dalam bentuk A(i,j).

- Memasukkan Elemen Matrik A

```
>> A = [ 2 4 6; 5 7 8; 9 8 7]
A =
     2     4     6
     5     7     8
     9     8     7
```

- Mengakses Elemen – Elemen Tertentu dari Matrik A

```
>> A(2,1), A(3,3), A(1,3), A(3,2)
ans =
     5
ans =
     7
ans =
     6
ans =
     8
```

- Cara lain memasukkan Matrik B

```
>> B(1,1) = 1, B(1,2) = 4, B(1,3)= 5, B(2,1) = 7, B(2,2)= 4,
B(2,3) = 5, B(3,1) = 1, B(3,2) = 8, B(3,3)= 4

B =
     1

B =
     1     4
```

```
B =
     1     4     5

B =
     1     4     5
     7     0     0

B =
     1     4     5
     7     4     0

B =
     1     4     5
     7     4     5

B =
     1     4     5
     7     4     5
     1     0     0

B =
     1     4     5
     7     4     5
     1     8     4
```

- Menampilkan Matrik B

```
>> B
B =
     1     4     5
     7     4     5
     1     8     4
```

- Menambah Elemen dan mengubah ukuran Matrik B

```
>> B(4,3) = 3
B =
     1     4     5
     7     4     5
     1     8     4
     0     0     3
```

- Mengubah Elemen Baris Ke-4 Pada Matrik B

```
>> B(4,:) = [1,2,3]
B =
     1     4     5
     7     4     5
```

```
     1     8     4
     1     2     3
```

- Mengubah Elemen Matrik B(3,2) dan B(3,3) dengan 1 dan 2

```
>> B(3,[2 3]) = [1,2]
B =
     1     4     5
     7     4     5
     1     1     2
     1     2     3
```

- Menampilkan Elemen Matrik B hanya baris ke 2

```
>> B(2,:)

ans =
     7     4     5
```

- Menampilkan Elemen Matrik B hanya kolom ke 2

```
>> B(:,2)
ans =
     4
     4
     4
     2
```

- Menampilkan Elemen Matrik B hanya baris 1 sampai 3

```
>> B(1:3,:)
ans =
     1     4     5
     7     4     5
     1     4     5
```

- Menampilkan Elemen Matrik B hanya baris 2 dan 3 dengan kolom 1 dan 2

```
>> B([2,3],1:2)
ans =
     7     4
     1     4
```

- Memasukkan Elemen – Elemen Matrik C

```
>> C = [1:4;5:8; 2 2 2 2]
C =
     1     2     3     4
```

```
     5     6     7     8
     2     2     2     2
```

- Mengcopi Matrik C ke Matrik D

```
>> D = C
D =
     1     2     3     4
     5     6     7     8
     2     2     2     2
```

- Menghapus Seluruh Elemen Kolom Ke 2 dari Matrik D

```
>> D(:,2) = []
D =
     1     3     4
     5     7     8
     2     2     2
```

- Menentukan Nilai Maksimum Elemen Matrik D

```
>> max(max(D))
ans =
     8
```

4.3.3 Operasi dan Fungsi Pada Matriks

Operasi dan fungsi pada matrik yang sering digunakan diperlihatkan pada Tabel 4.1 berikut

Tabel 4.1 Operasi dan Fungsi Matrik

Fungsi	Keterangan	Contoh
det	Fungsi ini berguna untuk menghitung determinan dari suatu matrik	det(a)
size	Fungsi ini berguna untuk menghitung ukuran matrik	size(a)
trace	Fungsi ini berguna untuk menghitung jumlah elemen diagonal matrik	trace (a)
norm	Fungsi ini berguna untuk menghitung panjang Euclidean vektor	norm(p)

+	Simbol ini digunakan untuk menjumlahkan matrik	C = A + B
-	Simbol ini digunakan untuk mengurangkan matrik	C = A – B
*	Simbol ini digunakan untuk mengalikan matrik	C = A * B
.*	Simbol ini digunakan untuk mengalikan elemen dengan elemen dengan syarat ketentuan memiliki ukuran yang sama	C = A.*B
^	Simbol ini digunakan untuk memangkatkan matrik dengan suatu skalar	C = A^k
.^	Simbol ini digunakan untuk memangkatkan elemen demi elemen matrik dengan skalar	C = A^.k
‘	Simbol ini digunakan untuk transpose matrik	A’
./	Simbol ini digunakan untuk membagi elemen per elemen dengan ketentuan memiliki ukuran yang sama	C = A./B
\	Simbol ini digunakan untuk menghasilkan solusi AX = B	C = A\B
/	Simbol ini digunakan untuk menhasilkan solusi XA = B	C = B/A
inv	Fungsi ini berguna untuk menghasilkan invers matrik dengan ketentuan invers matrik merupakan matrik bujursangkar	C= inv(D)
null	Fungsi ini berguna untuk menghasilkan orthonormal basis untuk spasi null dari matrik yang dihasilkan dari singular value decomposition (svd)	C = null(A)
orth	Fungsi ini berguna untuk menghasilkan orthonormal basis pada jangkauan A	C = orth(A)
rref	Fungsi ini berguna untuk menghasilkan reduce row echelon form dari matrik	C = rref(A)
eig	Fungsi ini berguna untuk menghasilkan suatu vektor yang berisi nilai eigen dari suatu matrik bujursangkar	C = eig(A)
svd	Fungsi ini berguna untuk menentukan nilai singular dari suatu matrik	P = svd(A)

linspace	Fungsi ini berguna untuk menghasilkan suatu vektor dengan nilai antara a dan b	X = linspace(a,b,n)
logspace	Fungsi ini berguna untuk menghasilkan suatu vektor yang dimulai dari 10^a dan berakhir pada 10^b sebanyak n nilai elemen.	X = logspace(a,b,n)
eye	Fungsi ini berguna untuk menghasilkan matrik identitas	A = eye(n)
zeros	Fungsi ini berguna untuk menghasilkan matrik dengan elemen – elemennya bernilai nol	A = zeros(n,m)
ones	Fungsi ini berguna untuk menghasilkan matrik dengan elemen – elemennya bernilai satu	A = ones(n,m)
diag	Fungsi ini berguna untuk menghasilkan diagonal matrik	A = diag(x)
tril	Fungsi ini berguna untuk menghasilkan matrik segitiga bawah dari sebuah matrik	X = tril(A)
triu	Fungsi ini berguna untuk menghasilkan matrik segitiga atas dari sebuah matrik	X = triu(A)
rand	Fungsi ini berguna untuk menghasilkan matrik dengan elemen – elemen terditribusi antara 0 dan 1 (default n = m)	A = rand(n,m)
randn	Fungsi ini berguna untuk menghasilkan matrik dengan elemen – elemen terdistribusi normal	A = randn(n,m)
max	Fungsi ini berguna untuk menentukan nilai maksimum dari elemen dari setiap kolom matrik atau nilai maksimum dari seluruh elemen jika merupakan vektor	S = max (A)
min	Fungsi ini berguna untuk menentukan nilai minimum dari elemen dari setiap kolom matrik atau nilai minimum dari seluruh elemen jika merupakan vektor	S = min(A)
sum	Fungsi ini berguna untuk menentukan nilai jumlah dari elemen dalam setiap kolom matrik atau nilai jumlah dari seluruh elemen jika merupakan vektor	S = sum(A)

Adapun contoh – contoh dari fungsi –fungsi Matlab sebagai berikut

```
>> A = [ 1 4 2; 4 5 6; 8 7 6]
A =
     1     4     2
     4     5     6
     8     7     6

>> P = det(A)
P =
    60

>> P1 = size(A)
P1 =
     3     3

>> P2 = trace(A)
P2 =
    12

>> B = [ 1  5  3; 3  4  2; 1 2 3]
B =
     1     5     3
     3     4     2
     1     2     3

>> C = [ 6 5 4 3 2 1]
C =
     6     5     4     3     2     1

>> norm(C)
ans =
    9.5394

>> P3 = A + B
P3 =
     2     9     5
     7     9     8
     9     9     9

>> P4 = A - B
P4 =
     0    -1    -1
     1     1     4
     7     5     3

>> P5 = A * B
P5 =
```

```
    15    25    17
    25    52    40
    35    80    56

>> P6 = A.*B
P6 =
     1    20     6
    12    20    12
     8    14    18

>> P7 = A^2
P7 =
    33    38    38
    72    83    74
    84   109    94

>> P8 = A.^2
P8 =
     1    16     4
    16    25    36
    64    49    36

>> P9 = A’
P9 =
     1     4     8
     4     5     7
     2     6     6
>> P10 = A./B
P10 =
    1.0000    0.8000    0.6667
    1.3333    1.2500    3.0000
    8.0000    3.5000    2.0000

>> P11 = A\B
P11 =
   -0.4667   -1.2000   -0.2333
   -0.0667    1.4000    0.9667
    0.8667    0.3000   -0.3167

>> P12 = B/A
P12 =
    1.2000    0.2500   -0.1500
    0.6000   -0.3333    0.4667
         0    0.7500   -0.2500

>> P13 = inv(A)
P13 =
```

```
   -0.2000   -0.1667    0.2333
    0.4000   -0.1667    0.0333
   -0.2000    0.4167   -0.1833

>> P14 = null(A)
P14 =
   Empty matrix: 3-by-0

>> D = [ 0 4; 4 2]
D =
     0     4
     4     2

>> P15 = null(D)
P15 =
   Empty matrix: 2-by-0

>> P16 = orth(A)
P16 =
   -0.2657   -0.7428   -0.6146
   -0.5595   -0.4004    0.7257
   -0.7851    0.5367   -0.3092

>> P17 = rref(A)
P17 =
     1     0     0
     0     1     0
     0     0     1

>> P18 = eig(A)
P18 =
  14.5512 + 0.0000i
  -1.2756 + 1.5799i
  -1.2756 - 1.5799i

>> x =linspace(1,7,10)
x =
Columns 1 through 9
1.0000    1.6667    2.3333    3.0000    3.6667    4.3333    5.0000
5.6667    6.3333

Column 10
7.0000

>> x = logspace(1,6,6)
x =
10          100         1000        10000       100000      1000000
```

```
> X = zeros(4)
X =
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0

>> Y = eye(4)
Y =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1

>> Y = ones(4)
Y =
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1

>> P20 = diag(A)
P20 =
     1
     5
     6

>> P21 = tril(A)
P21 =
     1     0     0
     4     5     0
     8     7     6

>> P22 = triu(A)
P22 =
     1     4     2
     0     5     6
     0     0     6

>> P23 = rand(2,4)
P23 =
    0.6557    0.8491    0.6787    0.7431
    0.0357    0.9340    0.7577    0.3922

>> P24 = randn(2,4)
P24 =
```

```
    0.2939    0.8884   -1.0689   -2.9443
   -0.7873   -1.1471   -0.8095    1.4384

>> P25 = max(A)
P25 =
     8     7     6

>> P27 = min(A)
P27 =
     1     4     2

>> P28 = sum(A)
P28 =
    13    16    14
```

4.4 Rangkuman

Pembahasan vektor dan matrik pada Matlab dilakukan secara terpisah. Vektor adalah kumpulan nilai yang dituliskan dalam satu dimensi 1xm sedangkan matrik adalah kumpulan nilai dalam dimensi m x m. Matrik dan vektor adalah jantung dari komputasi matlab. Vektor dan matrik digunakan untuk menyimpan list data atau signal. Vektor dan matrik dapat diberi nama dan diperlakukan seperti variabel -variabel lain pada matlab, namun bedanya operasi yang dilakukan oleh vektor dan matrik dilakukan pada tiap elemen satu demi satu.

4.5 Soal – Soal

Soal 4.1: Untuk matrik yang dinyatakan dalam persamaan (4.2) dan (4.3) berikut

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 1 \\ 3 & 2 & 1 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.2)$$

$$B = \begin{bmatrix} 4 & 4 & 5 \\ 6 & 1 & 2 \\ 3 & 5 & 5 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.3)$$

Dengan menggunakan Matlab tentukan

a. Determinan matrik A dan matrik B

b. Ukuran dari matrik A dan matrik B
c. Trace dari matrik A dan matrik B
d. Norm matrik A dan matrik B
e. C = A + B
f. C = A - B
g. C = A * B
h. C = A.* B
i. C = A^2
j. C = A.^2
k. C = A./B
l. C = A\B
m. C = B/A
n. C = null(A)
o. C = orth(A)
p. C = rref(A)
q. Transpose dari matrik A dan matrik B
r. Invers dari matrik A dan matrik B
s. Nilai eigen dari matrik A dan matrik B
t. Nilai singular dari matrik A dan matrik B
u. Matrik segitiga atas dari matrik A dan matrik B
v. Matrik segitiga bawah dari matrik A dan matrik B
w. Tentukan nilai maksimum dari elemen – elemen matrik A dan matrik B
x. Tentukan nilai minimum dari elemen – elemen matrik A dan matrik B
y. Tentukan jumlah kolom elemen – elemen matrik A dan matrik B
z. Tentukan diagonal dari matrik A dan matrik B

Soal 4.2: Dengan menggunakan Matlab, buatlah

a. Matrik identitas dengan dimensi 5 x 5
b. Matrik dengan dimensi 3 x 3 dengan elemen – elemen semuanya bernilai nol
c. Matrik dengan dimensi 4 x 4 dengan elemen – elemen semuanya bernilai satu
d. Matrik dengan dimensi 3 x 4 dimana elemen –elemen terditribusi antara 0 dan 1 (default n = m)
e. Matrik dengan dimensi 3 x 4 dimana elemen –elemen terditribusi normal

BAB V
STRUKTUR KONTROL PADA MATLAB

5.1 Pendahuluan

Bagian ini membahas struktur kontrol pada Matlab. Adapun struktur control yang dibahas meliputi operasi relasi dan logika pada Matlab, perintah *if*…., perintah *switch*, perintah *for*, peirntah *while*, perintah *continue*, perintah *break* dan perintah *return*.

5.2 Operator Relasi dan Logika Pada Matlab

Untuk menggunakan perintah arus kendali, perlu untuk melakukan operasi yang menghasilkan nilai dalam logika *true* atau *false*. Dalam bahasa Matlab, hasil dari operasi logika adalah 1 (satu) jika bernilai benar dan 0 (nol) jika bernilai salah. Operator relasi ini digunakan untuk membandingkan dua array yang memiliki ukuran yang sama atau suatu array dengan suatu skalar. Untuk operator relasi diperlihatkan pada Tabel 5.1 berikut

Tabel 5.1 Operator Relasi Pada Matlab

Operator	Keterangan
==	Sama dengan
~=	Tidak sama dengan
>	Lebih besar dari
<	Lebih kecil dari
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan

Operator Logika digunakan untuk kombinasi atau negasi dari operator relasi. Untuk operator logika diperlihatkan pada Tabel 5.2 berikut

Tabel 5.2 Operator Logika Pada Matlab

Operator	Keterangan
&	Dan
|	Atau
~	Negasi

Adapun contoh – contoh untuk operasi operator relasi dan operator logika diperlihatkan pada contoh 5.1 s/d contoh 5.5 berikut

Contoh 5.1: Pada contoh ini dilakukan operasi relasi untuk data yang bersifat skalar.

```
clc
clear all
close all
close all hidden
%
b = 1;
c = -2;
a1 = (b>c)
a2 = (b == c)
a3 = (b ~= c)
a4 = ~b
```

Hasil program

```
a1 =
     1
a2 =
     0
a3 =
     1
a4 =
     0
```

Contoh 5.2: Pada contoh ini dilakukan operasi relasi untuk data – data yang berbentuk vektor. Adapun data yang akan dibandingkan dinyatakan dalam bentuk persamaan (5.1) dan (5.2) berikut

$$x = [1.0000 \quad 6.0000 \quad 3.0000 \quad 10.0000 \quad 8.0000 \quad 0.0000 \quad 2.0000] \;\cdots\cdots\cdots \qquad (5.1)$$

$$y = [5.0000 \quad 2.0000 \quad 2.0000 \quad 6.0000 \quad 0.0000 \quad 0.0000 \quad 4.0000] \;\cdots\cdots \qquad (5.2)$$

Dengan menggunakan Matlab lakukan operasi relasi dan logika untuk data – data pada persamaan (5.1) dan (5.2)

Jawab :

Untuk data – data yang berbentuk vektor maka operasi relasi dan operasi logika dilakukan terhadap elemen – elemen vektor tersebut dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
x = [ 1.0000  6.0000  3.0000  10.0000  8.0000  0.0000
2.0000];
y = [ 5.0000  2.0000  2.0000   6.0000  0.0000  0.0000
4.0000];
%
x > y
%
y < x
%
x == y
%
x <= y
%
y >= x
%
x | y
%
x & (~y)
```

Hasil program

```
ans =
     0     1     1     1     1     0     0
ans =
     0     1     1     1     1     0     0

ans =
     0     0     0     0     0     1     0
ans =
     1     0     0     0     0     1     1
ans =
     1     0     0     0     0     1     1
ans =
     1     1     1     1     1     0     1
ans =
     0     0     0     0     1     0     0
```

Contoh 5.3: Pada contoh ini dilakukan operasi relasi untuk data – data yang berbentuk matrik. Adapun data yang akan dibandingkan dinyatakan dalam bentuk persamaan (5.3) berikut

$$x = \begin{bmatrix} 1 & 4 & 7 \\ 5 & 2 & 9 \\ 6 & 8 & 3 \end{bmatrix} \quad \ldots\ldots\ldots \quad (5.3)$$

Dengan menggunakan Matlab lakukan operasi relasi dan logika untuk data – data pada persamaan (5.3).

Jawab :

Untuk data – data yang berbentuk matrik, maka operasi relasi dan operasi logika dilakukan terhadap elemen – elemen matrik tersebut dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
x = [ 1.0000  4.0000  7.0000;  5.0000  2.0000  9.0000;  6.0000
8.0000  3.0000];
%
x == 0
%
x > 2
```

Hasil program

```
ans =
     0     0     0
     0     0     0
     0     0     0

ans =
     0     1     1
     1     0     1
     1     1     1
```

5.3 Perintah If……

Perintah if digunakan untuk mengambil keputusan instruksi yang harus dieksekusi berikutnya tergantung apakah ekspresi bernilai benar atau salah. Perintah if…… ini mempuyai berbagai bentuk dan variasi. Bentuk dan variasi dari perintah if…… diperlihatkan pada contoh 5.4 s/d 5.6.

Contoh 5.4: Pada bagian ini diperlihatkan contoh pengunaan perintah if…… dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = input('Nilai Ujian : ');
if (x > 75)
    disp('Anda Lulus')
end
disp('Anda Tidak Lulus')
```

Hasil program

```
Nilai Ujian : 70
Anda Tidak Lulus
```

Contoh 5.5: Contoh lain pengunaan perintah if…… dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = input('Nilai Ujian : ');
if (x > 75)
    disp('Anda Lulus')
else
    disp('Anda Tidak Lulus')
end
```

Hasil program

```
Nilai Ujian : 80
Anda                                                    Lulus
```

Contoh 5.6: Contoh lain pengunaan perintah if…… dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = input('Tinggi Badan  : ');
if (x > 190)
    disp('Kategori Sangat Tinggi')
elseif (x>170)
    disp('Kategori Tinggi')
elseif (x < 150)
    disp('Kategori Pendek')
else
    disp('Kategori Rata - Rata')
end
```

Hasil program

```
Tinggi Badan  : 198
Kategori Sangat Tinggi
```

5.4 Perintah *Switch*

Perintah *switch* juga dapat melakukan seleksi dari beberapa ekpresi termasuk untuk skalar maupun untuk string. Perintah *switch* ini dapat digunakan untuk menggantikan perintah seleksi if…...Untuk contoh perintah *switch* diperlihatkan pada contoh 5.7.

Contoh 5.7: Pada bagian ini diperlihatkan contoh pengunaan perintah *switch* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = input('Nilai Huruf  : ');
switch x
    case 'A'
        disp('Sangat Memuaskan')
    case 'B'
        disp('Memuaskan')
```

```
    case 'C'
        disp('Cukup')
    case 'D'
        disp('Jelek')
    case 'E'
        disp('Sangat Jelek')
    otherwise
        disp('Tidak Ada Dalam Daftar')
end
```

Hasil program

```
Nilai Huruf  : 'C'
Cukup
```

5.5 Perintah For

Perintah *for* digunakan untuk mengulang blok instruksi sebanyak jumlah tertentu. Perintah *for* ini mempuyai berbagai bentuk dan variasi. Bentuk dan variasi dari perintah *for* diperlihatkan pada contoh 5.8 s/d 5.10.

Contoh 5.8: Pada bagian ini diperlihatkan contoh pengunaan perintah *for* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
jumlah = 0;
for i = 1 : 15
    jumlah = jumlah + i;
end
jumlah
```

Hasil program

```
jumlah =
   120
```

Contoh 5.9: Contoh lain pengunaan perintah *for* dengan kode Matlab berikut

```
clc
clear all
close all
```

```
close all hidden
%
jumlah = 0;
for i = 1 : 2 : 15
   disp(i)
end
```

Hasil program

```
     1
     3
     5
     7
     9
    11
    13
    15
```

Contoh 5.10: Contoh lain pengunaan perintah *for* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
for i = 20 : -2 : 1
   disp(i);
end
```

Hasil program

```
    20
    18
    16
    14
    12
    10
     8
     6
     4
     2
```

5.6 Perintah While

Perintah *while* pada prinsipnya sama dengan perintah *for*, yang digunakan untuk mengulang blok perintah sepanjang ekspresi bernilai benar. Untuk contoh perintah *while* diperlihatkan diperlihatkan pada contoh 5.11 dan 5.12.

Contoh 5.11: Pada bagian ini diperlihatkan contoh pengunaan perintah *while* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
jumlah = 0;
i = 1;
while i <= 15
    jumlah = jumlah + i;
    i = i + 1;
end
jumlah
```

Hasil program

```
jumlah =
   120
```

Contoh 5.12: Contoh lain pengunaan perintah *while* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
k = 1;
while k <= 10
    disp('Padang')
    k = k + 1;
end
```

Hasil program

```
Padang
Padang
Padang
```

```
Padang
Padang
Padang
Padang
Padang
Padang
Padang
```

5.7 Perintah Continue

Perintah *continue* dapat digunakan untuk mengulang kembali dari awal *loop/ perulangan* sebelum kondisi yang menyebabkan mengulang kembali dari perulangan ditemukan. Untuk contoh perintah *continue* diperlihatkan pada contoh 5.13.

Contoh 5.13: Pada bagian ini diperlihatkan contoh pengunaan perintah *continue* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
jumlah = 0;
for k = 1:10;
    if (k== 4)
        continue
    else
        jumlah = jumlah + k;
    end
end
jumlah
```

Hasil program

```
jumlah =
    51
```

5.8 Perintah Break

Perintah *break* dapat digunakan untuk mengakhiri perulangan sebelum kondisi yang menyebabkan keluar perulangan ditemukan. Untuk contoh perintah *break* diperlihatkan pada contoh 5.14.

Contoh 5.14: Pada bagian ini diperlihatkan contoh pengunaan perintah *break* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
k = 1;
for k = 1:15;
    if (k == 12)
        break
    else
        disp(k);
    end
end
```

Hasil program

```
     1
     2
     3
     4
     5
     6
     7
     8
     9
    10
    11
```

5.9 Perintah Return

Perintah *return* digunakan untuk mengakhiri eksekusi perintah setelah ditemukannya perintah *return.* Untuk contoh perintah *break* diperlihatkan pada contoh 5.15.

Contoh 5.15 Pada bagian ini diperlihatkan contoh pengunaan perintah *return* dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
```

```
k = 1;
for k = 1:15;
    if (k == 8)
        return;
    else
        disp(k);
    end
end
```

Hasil program

```
1
2
3
4
5
6
7
```

5.10 Rangkuman

Matlab menyediakan fungsi untuk mengendalikan program. Adapun fungsi – fungsi yang berperan sebagai struktur control program diantarannya operator relasi dan logika, perintah *if*……, perintah *switch*, perintah *for*, perintah *while*, perintah *continue*, perintah *break* dan perintah *return.*

5.11 Soal – Soal

Soal 5.1: deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
b = -3;
c = 1;
a1 = (b>c)
a2 = (b == c)
a3 = (b ~= c)
a4 = ~b
```

Soal 5.2: Untuk data – data pada persamaan (5.4) dan (5.5) berikut

$$x = \begin{bmatrix} 2.0000 & 12.0000 & 6.0000 & 20.0000 & 16.0000 & 1.0000 & 4.0000 \end{bmatrix} \quad \ldots\ldots \quad (5.4)$$

$$y = \begin{bmatrix} 2.0000 & 1.0000 & 3.0000 & 2.0000 & 1.0000 & 0.0000 & 4.0000 \end{bmatrix} \quad \ldots\ldots \quad (5.5)$$

Dengan menggunakan Matlab, lakukan operasi relasi dan logika untuk data – data tersebut

Soal 5.3: Untuk matrik pada persamaan (5.6) berikut

$$x = \begin{bmatrix} 1 & 4 & 5 \\ 2 & 1 & 3 \\ 8 & 8 & 5 \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (5.6)$$

Dengan menggunakan Matlab, lakukan operasi relasi dan logika untuk elemen pada matrik tersebut

Soal 5.4: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
x = input('Nilai UTS : ');
if (x > 55)
    disp('Anda Lulus')
end
disp('Anda Harus Mengulang')
```

Soal 5.5: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
x = input('Nilai Ujian : ');
if (x > 55)
    disp('Anda Lulus')
else
    disp('Anda Harus Mengulang')
end
```

Soal 5.6: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
x = input('Berat Badan  : ');
if (x > 90)
    disp('Kategori Sangat Berat')
elseif (x > 80)
    disp('Kategori Berat')
elseif (x < 75)
    disp('Kategori Tidak Berat')
else
    disp('Kategori Rata - Rata')
end
```

Soal 5.7: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
x = input('Masukan Kota Tujuan Wisata (1...6)  : ');
switch x
    case '1'
        disp('Bukittingi')
    case '2'
        disp('Payakumbuh')
    case '3'
        disp('Batusangkar')
    case '4'
        disp('Solok Selatan')
    case '5'
        disp('Sawahlunto')
    otherwise
        disp('Padang')
end
```

Soal 5.8: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
```

```
%
jumlah = 0;
for i = 20 : 45
    jumlah = jumlah + i;
end
jumlah
```

Soal 5.9: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
jumlah = 0;
for i = 10 : 2 : 25
   disp(i)
end
```

Soal 5.10: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
for i = 40 : -2 : 5
   disp(i);
end
```

Soal 5.11: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
jumlah = 0;
i = 5;
while i <= 25
    jumlah = jumlah + i;
    i = i + 1;
end
jumlah
```

Soal 5.12: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
k = 1;
while k <= 20
    disp('Fakultas Teknik')
    k = k + 1;
end
```

Soal 5.13: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
jumlah = 0;
for k = 1:20;
    if (k== 5)
        continue
    else
        jumlah = jumlah + k;
    end
end
jumlah
```

Soal 5.14: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
k = 1;
for k = 1:25;
    if (k == 10)
        break
    else
        disp(k);
    end
end
```

Soal 5.15: Deskripsikan kode – kode Matlab berikut pada *Matlab Editor*

```
clc
clear all
close all
close all hidden
%
k = 1;
for k = 1:25;
    if (k == 10)
        return;
    else
        disp(k);
    end
end
```

BAB VI
PEMBACAAN DAN PENULISAN DATA PADA MATLAB

6.1 Pendahuluan

Pada bagian ini akan dibahas mengenai pembacaan dan penulisan data pada Matlab. Ada beberapa tipe pembacaan dan penulisan data pada Matlab diantaranya tipe .mat, tipe .txt dan tipe .xls. Pembahasan akan dimulai penulisan data pada Matlab dan diakhiri dengan pembacaan data pada Matlab.

6.2 File .mat

Pembacaan dan penulisan data pada Matlab untuk tipe .mat merupakan standard pembacaan dan penulisan data dengan Matlab. Untuk penulisan data tipe .mat digunakan fungsi **save** dan untuk pembacaan data tipe .mat digunakan fungsi **load.** Untuk contoh penulisan diperlihatkan pada contoh berikut

Contoh 6.1: Bentuk file data.mat dengan kode berikut

```
clc
clear all
close all
close all hidden
%
x = 20;
y = 5;
z = {10,'string'};
save data x y z
```

Hasil program

Hasil dari program diperoleh file dengan nama **data.mat**

Untuk pembacaan data dilakukan dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
load data.mat
x
y
z
```

Hasil program

```
x =
   20
y =
    5
z =
   [10]   'string'
```

Contoh 6.2: Bentuk file data.mat dengan kode berikut

```
clc
clear all
close all
close all hidden
%
p = input('Panjang Persegi Panjang : ');
l = input('Lebar Persegi Panjang   : ');
luas = p*l;
save hasil_hitung p l luas
```

Hasil program

Hasil dari program diperoleh file dengan nama **hasil_hitung.mat**

Untuk pembacaan data dilakukan dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
load hasil_hitung
p
l
luas
```

Hasil program

```
p =
    50
l =
    10
luas =
   500
```

Contoh 6.3: Bentuk file data.mat dengan kode berikut

```
clc
clear all
close all
close all hidden
%
r = rand(10,5)
save data_random r
```

Hasil program
Hasil dari program diperoleh file dengan nama **data_random.mat**
Untuk pembacaan data dilakukan dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
load data_random
r
```

Hasil program

```
r =
    0.3257    0.1080    0.9585    0.7517    0.8776
    0.6302    0.4599    0.7900    0.3684    0.0144
    0.2303    0.4509    0.4519    0.9418    0.2943
    0.5799    0.5511    0.3334    0.0172    0.1799
    0.6032    0.8054    0.0591    0.8291    0.9263
    0.5999    0.7009    0.7409    0.6266    0.0682
    0.4484    0.8722    0.5068    0.5387    0.5811
    0.0354    0.0522    0.1999    0.6505    0.6372
    0.5138    0.2197    0.4272    0.7266    0.6513
    0.4077    0.4596    0.1687    0.0945    0.8646
```

6.3 File .txt

Pembacaan dan penulisan data pada Matlab untuk tipe .txt dilakukan untuk data yang berbentuk teks biasa. Adapun fungsi yang digunakan yang digunakan untuk pembacaan dan penulisan data untuk tipe .txt dilakukan dalam bentuk [fid] = **fopen(*filename*, *permissions*)** dimana *filename* merupakan nama file tempat penyimpanan data dan *permissions* merupakan fitur – fitur dari file tempat penyimpanan data tersebut. Beberapa fitur –fitur dari *permissions* diperlihatkan pada Tabel 6.1 berikut

Tabel 6.1 Fitur – Fitur Permission

Permission	Fungsi
‘r’	Untuk membaca data dari file
‘w’	Untuk menulis data
‘a’	Untuk membuat file, menulis data dan menambahkan data terakhir pada file
‘r+’	Untuk membaca dan menuliskan data
‘w+’	Untuk membuat file, menulis dan membaca data
‘a+’	Untuk membuat file, menulis dan membaca data serta menambahkan data terakhir pada file
‘W’	Untuk membuka file dan menulis data tanpa sama rata
‘A’	Untuk membuka file dan menambahkan data tanpa sama rata

Adapun contoh pembacaan dan penulisan data dengan tipe .txt diperlihatkan pada contoh 6.4 s/d contoh 6.6.

Contoh 6.4: Bentuk file data.txt dengan kode berikut

```
clc
clear all
close all
close all hidden
%
fid = fopen(‘data.txt’,’w+’);
for i = 1:10
    fprintf(fid,’%d  %d %d\n’,i, i^2, i^3);
end
fclose(fid);
```

Hasil program
Hasil dari program diperoleh file dengan nama **data.txt.** Adapun isi dari file **data.txt** ini diperlihatkan sebagai berikut

```
1  1 1
2  4 8
3  9 27
4  16 64
5  25 125
6  36 216
```

```
7  49 343
8  64 512
9  81 729
10 100 1000
```

Contoh 6.5: Bentuk file data_hitung.txt dengan kode berikut

```
clc
clear all
close all
close all hidden
%
fid = fopen('data_hitung.txt','w+');
for p = 10 : 15
    for l = 1 : 5
fprintf(fid,'%d  %d %d\n',p, l, p*l);
    end
end
fclose(fid);
```

Hasil program

Hasil dari program diperoleh file dengan nama **data_hitung.txt.** Adapun isi dari file **data_hitung.txt** ini diperlihatkan sebagai berikut

```
10  1 10
10  2 20
10  3 30
10  4 40
10  5 50
11  1 11
11  2 22
11  3 33
11  4 44
11  5 55
12  1 12
12  2 24
12  3 36
12  4 48
12  5 60
13  1 13
13  2 26
13  3 39
13  4 52
13  5 65
14  1 14
14  2 28
14  3 42
```

```
14   4 56
14   5 70
15   1 15
15   2 30
15   3 45
15   4 60
15   5 75
```

Contoh 6.6: Bentuk file data_suhu.txt dengan kode berikut

```
clc
clear all
close all
close all hidden
%
fid = fopen('data_suhu.txt','w+');
for C = 1 : 15
R = 0.8000*C;
F = (1.8000*C + 32.0000);
fprintf(fid,'%8.4g %8.4g %8.4g\n',C, R, F);
end
fclose(fid);
```

Hasil program

Hasil dari program diperoleh file dengan nama **data_suhu.txt.** Adapun isi dari file **data_suhu.txt** ini diperlihatkan sebagai berikut

```
       1       0.8      33.8
       2       1.6      35.6
       3       2.4      37.4
       4       3.2      39.2
       5         4        41
       6       4.8      42.8
       7       5.6      44.6
       8       6.4      46.4
       9       7.2      48.2
      10         8        50
      11       8.8      51.8
      12       9.6      53.6
      13      10.4      55.4
      14      11.2      57.2
      15        12        59
```

6.4 File .xls

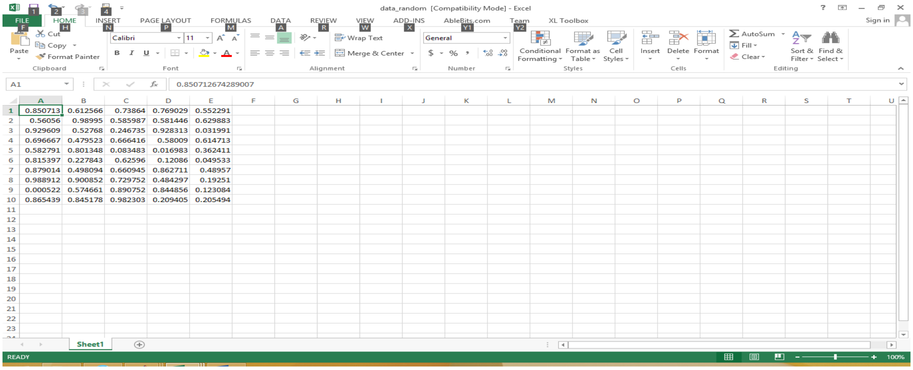
Pembacaan dan penulisan data pada Matlab untuk tipe .xls bisa dilakukan untuk data yang berukuran besar. Adapun fungsi yang digunakan yang digunakan untuk pembacaan dan penulisan data untuk tipe .xls dilakukan dalam bentuk `[success,message] = xlswrite(file,array,sheet,range)` dimana *file* merupakan nama file tempat penyimpanan data, jika file belum ada maka akan dibentuk dalam file direktori. *Array* berbentuk *double array* atau *cell array*. Setiap elemen dari *array* akan ditulis pada satu sel dalam lembar kerja Microsoft Excel. *Sheet* adalah nama lembaran dimana data akan ditulis dan jika sheet ini tidak didefinisikan maka akan dibentuk nama *sheet* tersebut. *Range* digunakan untuk mendiskripsikan lokasi data – data pada lembaran kerja yang akan ditulis. Adapun contoh pembacaan dan penulisan data dengan tipe .txt diperlihatkan pada contoh 6.7 s/d contoh 6.9.

Contoh 6.7: Bentuk file data_random.xls dengan kode berikut

```
clc
clear all
close all
close all hidden
%
A = rand(10,5);
[s] = xlswrite('data_random.xls',A)
```

Hasil program

Hasil dari program diperoleh file dengan nama **data_random.xls.** Adapun isi dari file **data_random.xls** ini diperlihatkan pada Gambar 6.1 berikut



Gambar 6.1 Data Format Excel

Untuk pembacaan data dilakukan dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
A = xlsread('data_random.xls','sheet1','A1:E5')
```

Hasil program

```
A =
    0.8507    0.6126    0.7386    0.7690    0.5523
    0.5606    0.9900    0.5860    0.5814    0.6299
    0.9296    0.5277    0.2467    0.9283    0.0320
    0.6967    0.4795    0.6664    0.5801    0.6147
    0.5828    0.8013    0.0835    0.0170    0.3624
```

6.5 Rangkuman

Ada beberapa cara pembacaan dan penulisan data pada Matlab. Pembacaan dan penulisan data pada Matlab diantaranya tipe .mat, tipe .txt dan tipe .xls. Untuk penulisan data tipe .mat merupakan standard pembacaan dan penulisan data dengan Matlab**.** Pembacaan dan penulisan data pada Matlab untuk tipe .txt dilakukan untuk data yang berbentuk teks biasa. Pembacaan dan penulisan data pada Matlab untuk tipe .xls bisa dilakukan untuk data yang berukuran besar.

6.6 Soal - Soal

Soal 6.1: Dengan menggunakan Matlab, lakukan pembentukkan file **data.mat** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
Q = 25;
R = 28;
S = {20,'string'};
save data Q R S
```

Soal 6.2: Dengan menggunakan Matlab, lakukan pembacaan file **data.mat** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
load data.mat
Q
R
S
```

Soal 6.3: Dengan menggunakan Matlab, lakukan pembentukkan file **hasil_hitung.mat** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
a = input('Alas Segitiga   : ');
t = input('Tinggi Segitiga    : ');
luas = 0.5000 * a * t;
save hasil_hitung a t luas
```

Soal 6.4: Dengan menggunakan Matlab, lakukan pembacaan file **hasil_hitung.mat** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
load hasil_hitung
a
t
luas
```

Soal 6.5: Dengan menggunakan Matlab, lakukan pembacaa file **data_random.mat** dengan kode berikut

```
clc
clear all
close all
close all hidden
```

```
%
r = randn(5,5)
save data_random r
```

Soal 6.6: Dengan menggunakan Matlab, lakukan pembacaan file **data_random.mat** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
load data_random
r
```

Soal 6.7: Dengan menggunakan Matlab, lakukan pembentukkan file **data.txt** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
fid = fopen('data.txt','w+');
for i = 1:5
    fprintf(fid,'%d  %d %d\n',i, i^4, i^5);
end
fclose(fid);
```

Soal 6.8: Dengan menggunakan Matlab, lakukan pembentukkan file **data_hitung.txt** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
fid = fopen('data_hitung.txt','w+');
for p = 20 : 25
    for l = 1 : 5
fprintf(fid,'%d  %d %d\n',p, l, p*l);
    end
end
fclose(fid);
```

Soal 6.8: Dengan menggunakan Matlab, lakukan pembentukkan file **data_suhu.txt** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
fid = fopen('data_suhu.txt','w+');
for C = 1 : 20
R = 0.8000*C;
F = (1.8000*C + 32.0000);
fprintf(fid,'%8.4g %8.4g %8.4g\n',C, R, F);
end
fclose(fid);
```

Soal 6.9: Dengan menggunakan Matlab, lakukan pembentukkan file **data_random.xls** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
B = randn(15,15);
[s] = xlswrite('data_random.xls',B)
```

Soal 6.10: Dengan menggunakan Matlab, lakukan pembacaan file **data_random.xls** dengan kode berikut

```
clc
clear all
close all
close all hidden
%
A = xlsread('data_random.xls','sheet1','A1:E5')
```

BAB VII
VISUALISASI PADA MATLAB

7.1 Pendahuluan

Pada bagian ini dibahas tentang visualisasi dengan menggunakan Matlab. Adapun pembahasan yang dilakukan meliputi visualisasi gambar 2 dimensi, visualisasi gambar 3 dimensi, visualisasi beberapa fungsi dalam satu gambar. Pembahasan diakhiri dengan rangkuman dan soal – soal latihan.

7.2 Visualisasi Gambar 2 Dimensi

Beberapa contoh visualisasi gambar 2 dimensi diperlihatkan dengan beberapa contoh. Untuk contoh visualisasi pertama diperlihatkan pada contoh 7.1 berikut

Contoh 7.1: Dengan menggunakan Matlab, plot data – data pada Tabel 7.1 berikut

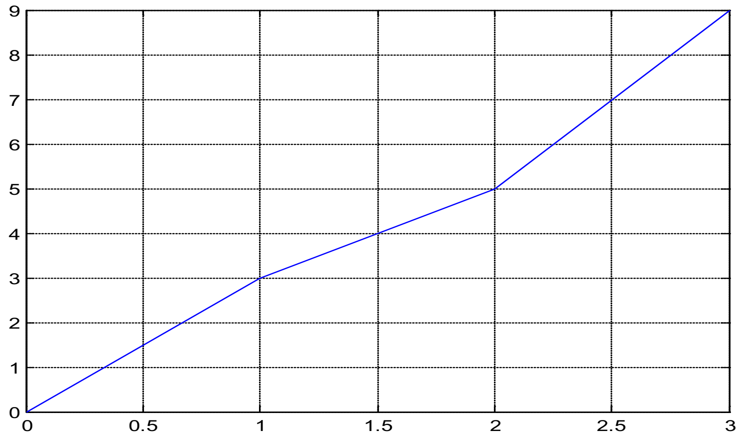
Tabel 7.1 Data – Data Hasil Pengukuran

x	0.0000	1.0000	2.0000	3.0000
y	0.0000	3.0000	5.0000	9.0000

Dengan menggunakan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = [ 0.0000   1.0000    2.0000   3.0000];
y = [ 0.0000   3.0000    5.0000   9.0000];
plot(x,y)
grid on
```

Hasil program memperlihatkan tampilan pada Gambar 7.1 berikut



Gambar 7.1 Tampilan Data – Data Pada Tabel 7.1

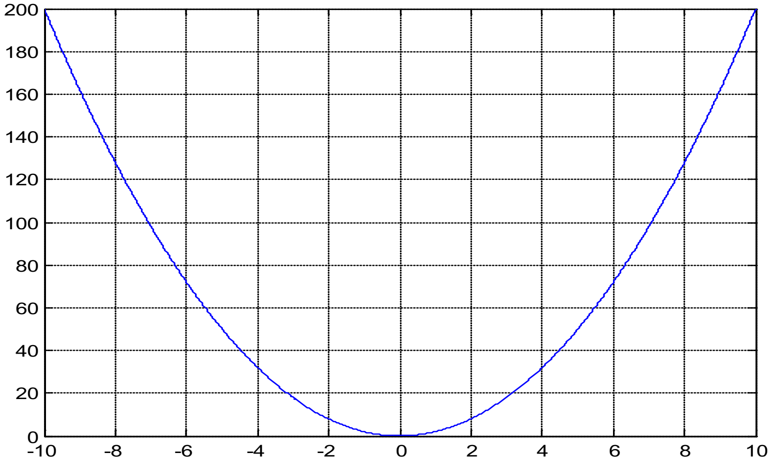
Contoh 7.2: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.1) berikut

$$f(x) = 2.0000x^2 \quad \text{.......} \quad (7.1)$$

Untuk $-10.0000 \leq x \leq 10.0000$**.** Adapun Persamaan (7.1) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = linspace(-10,10,1000)
plot(x,2*x.^2)
grid on
```

Hasil program



Gambar 7.2 Tampilan Grafik Persamaan (7.1)

Selain itu variasi garis yang digunakan pada visualisasi Gambar 2 dimensi dapat diubah style dan warnanya dengan argument seperti yang diperlihatkan pada Tabel 7.2 berikut

Tabel 7.2 Argumen Style dan Warna

Warna	Data Point	Style
b (blue)	point (.)	solid (-)
g (green)	Circle(o)	dotted (:)
r (red)	x-mark (x)	dashdot(-.)
c(cyan)	plus (+)	dashed(--)
m (magenta)	star (*)	no line
y (yellow)	square (s)	
k (black)	diamond (d)	
	triangle down (v)	
	triangle up (^)	
	triangle left (<)	
	triangle right (>)	
	pentagram (p)	
	hexagram (h)	

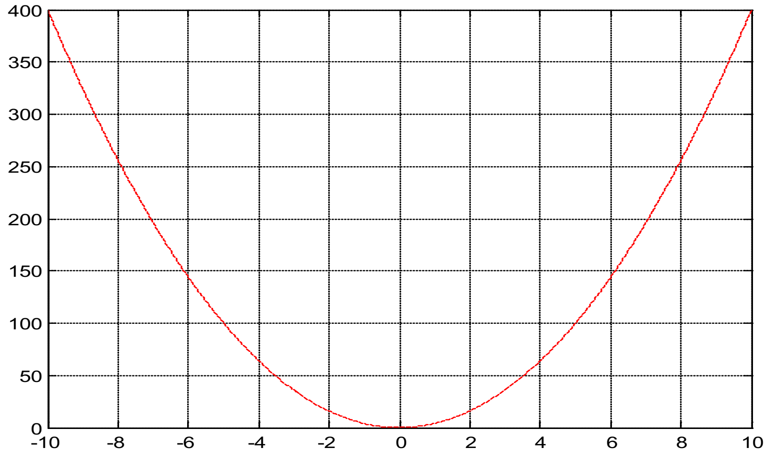
Contoh 7.3: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.2) berikut

$$f(x) = 4.0000x^2 \quad \text{.......} \quad (7.2)$$

Untuk $-10.0000 \le x \le 10.0000$ dengan warna garis merah dan putus - putus. Adapun Persamaan (7.2) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = linspace(-10,10,1000);
plot(x,4*x.^2,'r--')
grid on
```

Hasil program



Gambar 7.3 Tampilan Grafik Persamaan (7.2)

Contoh 7.4: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.3) dan (7.4) berikut

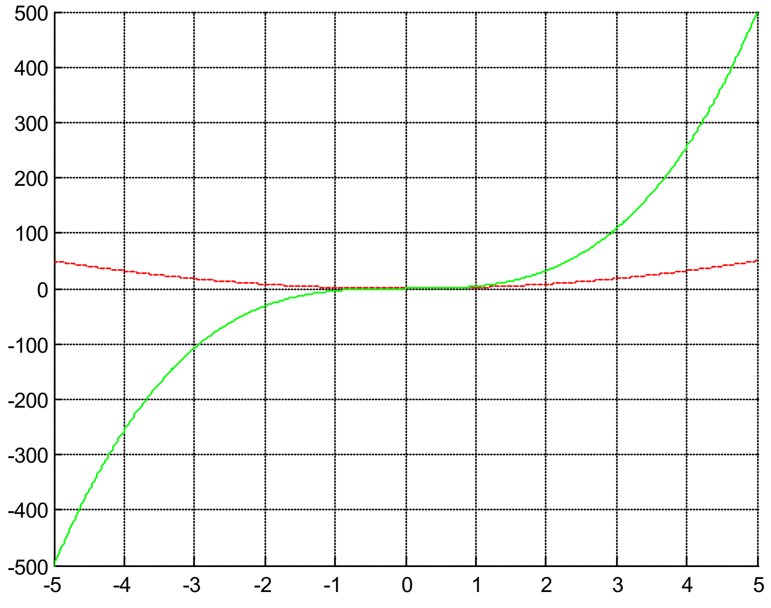
$$f_1(x) = 2.0000x^2 \quad \text{.......} \quad (7.3)$$

$$f_2(x) = 4.0000x^3 \quad \text{.......} \quad (7.4)$$

Untuk -5.0000 ≤ x ≤ 5.0000 dengan warna yang berbeda dan putus - putus. Adapun Persamaan (7.2) dan (7.3) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = linspace(-5,5,1000);
hold on
plot(x,2*x.^2,'r--')
plot(x,4*x.^3,'g-')
grid on
hold off
```

Hasil program



Gambar 7.4 Tampilan Grafik Persamaan (7.3) dan (7.4)

Contoh 7.5: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.5) dan (7.6) berikut

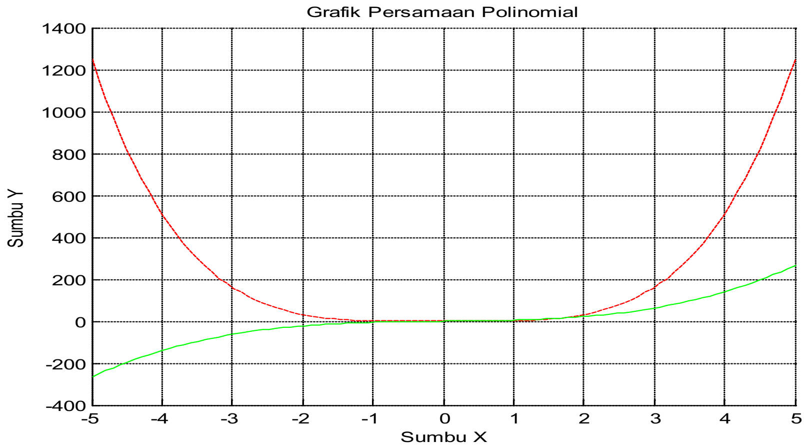
$$f_1(x) = 2.0000x^4 \quad \text{(7.5)}$$

$$f_2(x) = 2.0000x^3 + 3.0000x \quad \text{(7.6)}$$

Untuk $-10.0000 \leq x \leq 10.0000$ dengan warna yang berbeda dan putus – putus Adapun Persamaan (7.5) dan (7.6) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = linspace(-5,5,100);
hold on
plot(x,2*x.^4,'r--')
plot(x,(2*x.^3) + (3*x),'g-')
grid on
hold off
title('Grafik Persamaan Polinomial')
xlabel('Sumbu X')
ylabel('Sumbu Y')
```

Hasil program



Gambar 7.5 Tampilan Grafik Persamaan (7.5) dan (7.6)

7.3 Visualisasi Gambar 3 Dimensi

Untuk contoh visualisasi gambar 3 dimensi diperlihatkan dengan beberapa contoh. Untuk contoh visualisasi pertama diperlihatkan pada contoh 7.6 berikut

Contoh 7.6: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.7) s/d (7.9) berikut

$$x = \sin(t) \quad \text{..........} \quad (7.7)$$

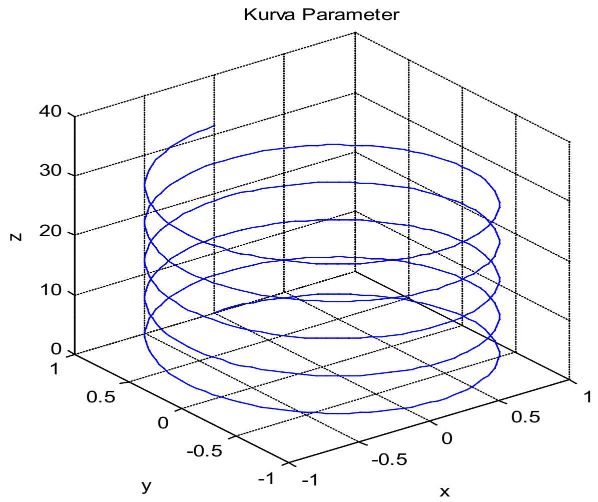
$$y = \sin(t) \quad \text{..........} \quad (7.8)$$

$$z = t \quad \text{..........} \quad (7.9)$$

Untuk $0 \le t \le 10.0000$**.** Adapun Persamaan (7.7) s/d (7.9) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
t = [0 :pi/50: 10*pi];
plot3(sin(t),cos(t),t)
grid on
axis square
title('Kurva Parameter')
xlabel('x')
ylabel('y')
zlabel('z')
```

Hasil program



Gambar 7.6 Tampilan Grafik Persamaan (7.7) s/d (7.9)

Contoh 7.7: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.10) berikut

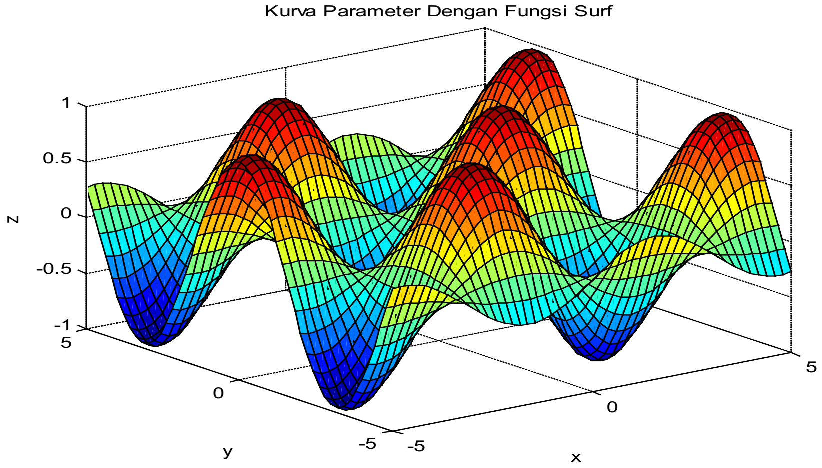
$$f(x, y) = \sin(x).\cos(y) \quad \text{..} \quad (7.10)$$

dengan batasan nilai -5.0000 $\leq x \leq$ 5.0000 dan -5.0000 $\leq y \leq$ 5.0000 dengan menggunakan fungsi *surf*. Adapun Persamaan (7.10) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = -5.0000 : 0.2000 : 5.0000;
y = -5.0000 : 0.2000 : 5.0000;
[X,Y] = meshgrid(x,y);
Z = sin(X).*cos(Y);
surf(X,Y,Z)
title('Kurva Parameter Dengan Fungsi Surf')
xlabel('x')
```

```
ylabel('y')
zlabel('z')
```

Hasil program



Gambar 7.7 Tampilan Grafik Persamaan (7.10)

Contoh 7.8: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.11) berikut

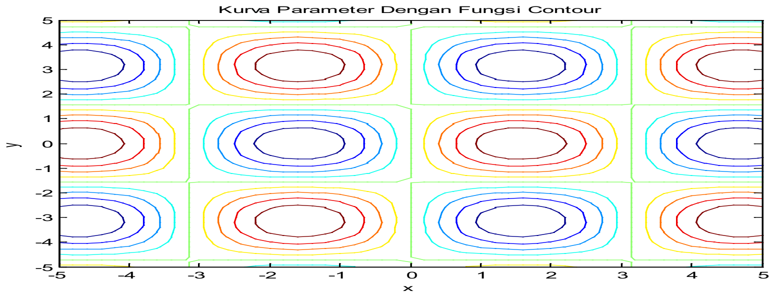
$$f(x, y) = \sin(x).\cos(y) \quad \ldots\ldots \quad (7.11)$$

dengan batasan nilai-5.0000 $\leq$ x $\leq$ 5.0000 dan -5.0000 $\leq$ y $\leq$ 5.0000 dengan menggunakan fungsi *contour*. Adapun Persamaan (7.11) digambarkan dengan kode Matlab berikut

```
clc
clear all
close all
close all hidden
%
x = -5.0000 : 0.2000 : 5.0000;
y = -5.0000 : 0.2000 : 5.0000;
[X,Y] = meshgrid(x,y);
Z = sin(X).*cos(Y);
contour(X,Y,Z)
title('Kurva Parameter Dengan Fungsi Contour')
xlabel('x')
```

```
ylabel('y')
zlabel('z')
```

Hasil program



Gambar 7.8 Tampilan Grafik Persamaan (7.11)

7.4 Visualisasi Beberapa Fungsi Dalam Satu Gambar

Selain itu Matlab juga mempuyai fasilitas untuk visualisasi beberapa fungsi dalam satu Gambar. Adapun contoh – contoh visualisasi beberapa fungsi dalam satu gambar diperlihatkan pada Contoh 7.9, Contoh 7.10 dan Contoh 7.11.

Contoh 7.9: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.12) dan (7.13) berikut

$$y_1 = \sin(2x) \quad \text{......} \quad (7.12)$$

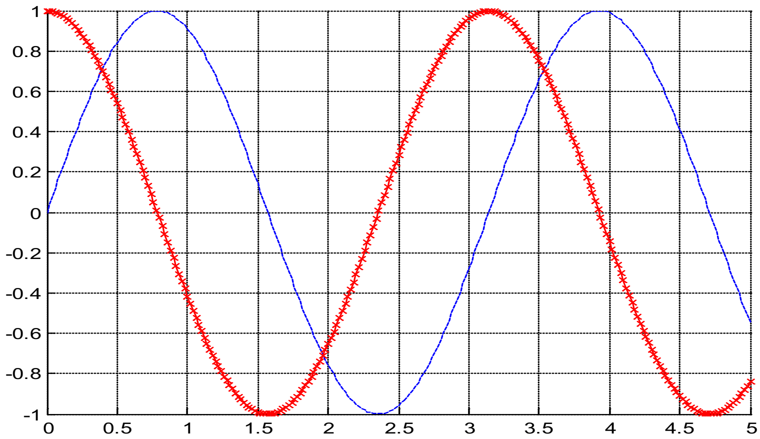
$$y_2 = \cos(2x) \quad \text{......} \quad (7.13)$$

dalam satu gambar yang sama. Adapun kode Matlab yang digunakan untuk plot fungsi pada persamaan (7.12) dan (7.13) berikut

```
clc
clear all
close all
close all hidden
%
n = 50;
x = 0 : 1/n : 5;
y1 = sin(2*x);
y2 = cos(2*x);
hold on
```

```
plot(x,y1,'b--');
plot(x,y2,'r-x');
grid on
hold off
```

Hasil program



Gambar 7.9 Tampilan Grafik Persamaan (7.12) dan (7.13)

Contoh 7.10: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.14) dan (7.16) berikut

$$y_1 = 2\sin(4x) \quad \text{..........} \quad (7.14)$$

$$y_2 = 2\cos(4x) \quad \text{..........} \quad (7.15)$$

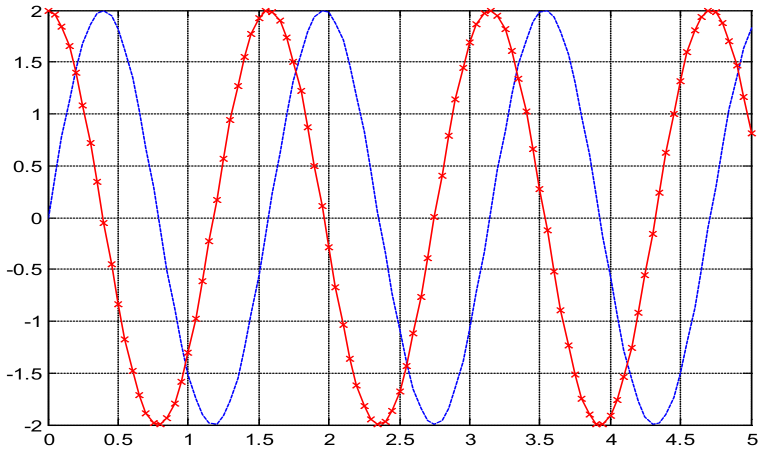
$$y_3 = 2\sin(5x) \quad \text{..........} \quad (7.16)$$

dalam satu gambar yang sama. Adapun kode Matlab yang digunakan untuk plot fungsi pada persamaan (7.14) dan (7.15) berikut

```
clc
clear all
close all
close all hidden
%
n = 20;
x = 0 : 1/n : 5;
y1 = 2 * sin(4*x);
```

```
y2 = 2 * cos(4*x);
y3 = 2 * sin(5*x);
plot(x,y1,'b--',x,y2,'r-x',x,y3,'k-');
grid on
```

Hasil program



Gambar 7.10 Tampilan Grafik Persamaan (7.14) s/d (7.16)

Contoh 7.11: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.17) s/d (7.20) berikut

$$y_1 = 8\sin(4x) \quad \text{.........} \quad (7.17)$$

$$y_2 = \cos(5x) \quad \text{.........} \quad (7.18)$$

$$y_3 = 2\sin(4x)\cos(3x) \quad \text{.........} \quad (7.19)$$

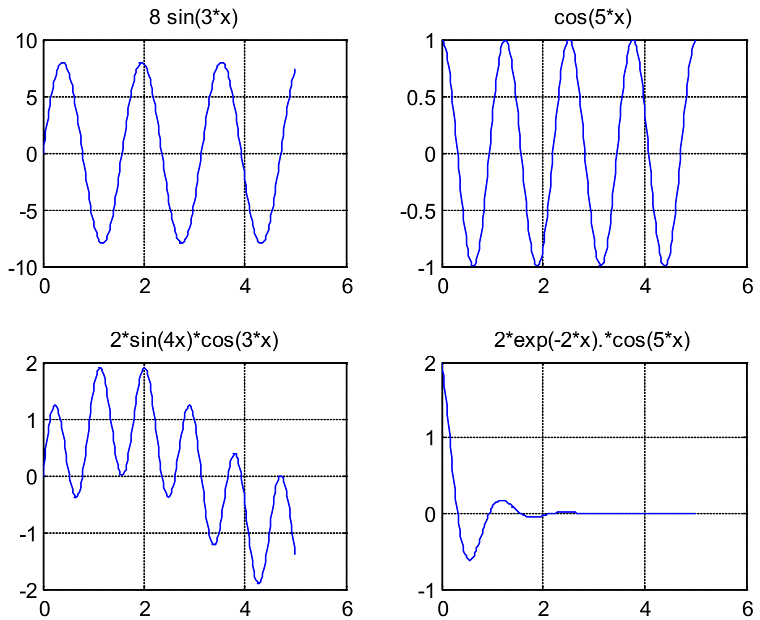
$$y_4 = 2e^{-2x}\cos(5x) \quad \text{.........} \quad (7.20)$$

dalam satu gambar yang sama. Adapun kode Matlab yang digunakan untuk plot fungsi pada persamaan (7.17) s/d (7.20) berikut

```
clc
clear all
close all
close all hidden
%
x = 0 : 0.02 : 5;
```

```
y1 = 8 * sin(4*x);
y2 = cos(5*x);
y3 = 2 * sin(4*x).*cos(3*x);
y4 = 2*exp(-2*x).*cos(5*x);
subplot(2,2,1)
plot(x,y1)
title('8 sin(3*x)')
grid on
subplot(2,2,2)
plot(x,y2)
title('cos(5*x)')
grid on
subplot(2,2,3)
plot(x,y3)
title('2*sin(4x)*cos(3*x)')
grid on
subplot(2,2,4)
plot(x,y4)
title('2*exp(-2*x).*cos(5*x)')
grid on
```

Hasil Program



Gambar 7.11 Tampilan Grafik Persamaan (7.17) s/d (7.20)

7.5 Rangkuman

Matlab dapat digunakan untuk menvisualisasi hasil. Visualisasi yang bisa dilakukan meliputi visualisasi gambasr 2 dimensi dan visualisasi 3 dimensi. Selain Matlab juga menyediakan fasilitas untuk visualisasi beberapa fungsi dalam satu gambar.

7.6 Soal - Soal

Soal 7.1: Dengan menggunakan Matlab, plot data – data hasil pengukuran yang diperlihatkan pada Tabel 7.3 berikut

Tabel 7.3 Data – Data Hasil Pengukuran

x	0.0000	1.0000	2.0000	3.0000
y	0.0000	10.0000	20.0000	30.0000

Soal 7.2: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.21) berikut

$$f(x) = x^2 + 2.0000 \quad \text{.......} \quad (7.21)$$

Untuk $-5.0000 \le x \le 5.0000$.

Soal 7.3: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.22) berikut

$$f(x) = x^3 \quad \text{.......} \quad (7.22)$$

Untuk $-5.0000 \le x \le 5.0000$ dengan warna garis biru dan putus - putus.

Soal 7.4: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.23) dan (7.24) berikut

$$f_1(x) = 4.0000x^3 \quad \text{.......} \quad (7.23)$$

$$f_2(x) = x^4 + 2.0000x^2 + 3.0000 \quad \text{.......} \quad (7.24)$$

Untuk $-5.0000 \le x \le 5.0000$ dengan warna yang berbeda dan putus - putus.

Soal 7.5: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.25) dan (7.26) berikut

$$f_1(x) = 2.0000x^2 \quad (7.25)$$

$$f_2(x) = 4.0000x^2 + 1.5000x \quad (7.26)$$

Untuk $-5.0000 \le x \le 5.0000$ dengan warna yang berbeda dan putus – putus

Soal 7.6: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.27) s/d (7.29) berikut

$$x = \sin(2t) \quad (7.27)$$

$$y = \sin(2t) \quad (7.28)$$

$$z = 4t \quad (7.29)$$

Untuk $0 \le t \le 20\pi$**.**

Soal 7.7: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.30) berikut

$$f(x, y) = 2\sin(2x).\cos(3y) \quad (7.30)$$

dengan batasan nilai $-5 \le x \le 5$ dan $-5 \le y \le 5$ dengan menggunakan fungsi *surf* dan *contour*

Soal 7.8: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.31) dan (7.32) berikut

$$y_1 = 2\sin(4x) \quad (7.31)$$

$$y_2 = 2\cos(4x) \quad (7.32)$$

dalam satu gambar yang sama dengan 3 cara yang berbeda.

Soal 7.9: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.33) s/d (7.36) berikut

$$y_1 = 4\sin(2x) \quad \text{.........} \quad (7.33)$$

$$y_2 = 2\sin(7x) \quad \text{.........} \quad (7.34)$$

$$y_3 = 2\cos(3x)\sin(4x) \quad \text{.........} \quad (7.35)$$

$$y_4 = 4e^{-4x}\sin(6x) \quad \text{.........} \quad (7.36)$$

dalam satu gambar yang sama dengan 3 cara yang berbeda.

Soal 7.10: Dengan menggunakan Matlab, plot fungsi pada persamaan (7.37) s/d (7.40) berikut

$$y_1 = \sin(2x) \quad \text{.........} \quad (7.37)$$

$$y_2 = \cos(6x) \quad \text{.........} \quad (7.38)$$

$$y_3 = 2\cos(3x)\sin(4x) \quad \text{.........} \quad (7.39)$$

$$y_4 = 4e^{-2x}\sin(6x) \quad \text{.........} \quad (7.40)$$

dalam satu gambar yang sama dengan 3 cara yang berbeda dan tampilan hasil yang berbeda.

BAB VIII
AKAR – AKAR PERSAMAAN

8.1 Pendahuluan

Pada bagian ini dibahas tentang penentuan akar –akar persamaan dari fungsi yang bersifat tidak linear. Penentuan akar - akar persamaan dilakukan dengan menggunakan metoda tertutup dan metoda terbuka. Untuk pembahasan metoda tertutup meliputi metoda grafis, metoda bagi dua dan metoda posisi palsu. Untuk metoda terbuka terdiri dari metoda iterasi satu titik sederhana, metoda Newton Raphson dan metoda Secant.

8.2 Metoda Tertutup

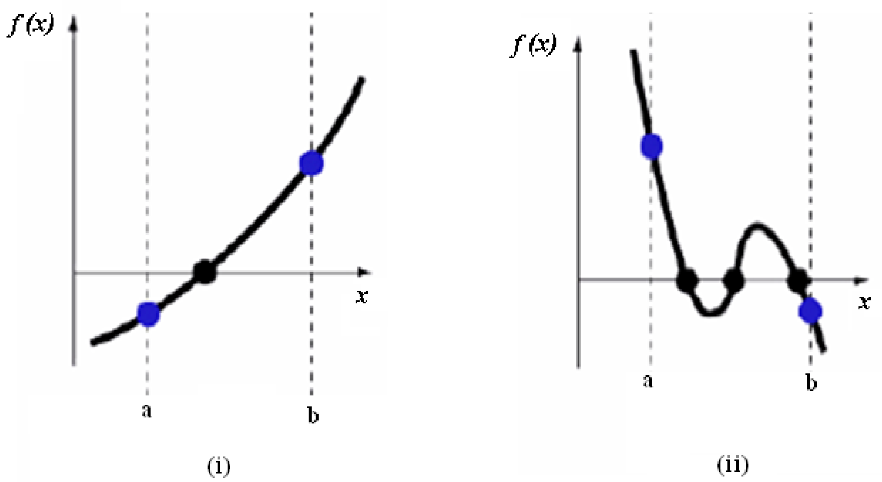
Pada bagian ini membahas tentang metoda – metoda yang memanfaatkan kenyataan bahwa suatu fungsi secara khas berganti tanda di sekitar suatu akar. Metoda – metoda tersebut dikenal sebagai metoda tertutup karena diperlukan dua tebakan awal untuk akar. Seperti yang tersirat oleh namanya, terkaan ini harus mengurung atau berada pada kedua sisi dari akar. Selain itu metoda yang termasuk ke dalam golongan ini mencari akar di dalam selang batas bawah dan batas atas. Selang batas bawah dan batas atas sudah dipastikan berisi minimal satu buah akar karena metoda jenis ini selalu berhasil menemukan akar. Dengan kata lain iterasinya selalu konvergen menuju akar. Pembahasan metoda tertutup akan dimulai dengan pembahasan metoda grafis, metoda bagi dua dan metoda posisi palsu.

8.2.1. Metoda Grafis

Metoda yang sederhana untuk memperoleh taksiran atas akar persamaan F(x) = 0 adalah membuat gambar grafik fungsi dan mengamati dimana ia memotong sumbu x. Titik ini yang mewakili nilai x dimana F(x) = 0. Akar-akar suatu persamaan dari suatu fungsi x sebenarnya adalah harga x yang membuat F(x) = 0. Selain itu metoda grafis sebagai bagian dari metoda tertutup tentu memerlukan selang [a,b] yang mengandung akar – akar dari persamaan F(x). Sebagaimana namanya, selang tersebut mengurung akar sejati. Aturan yang dipakai adalah mengurangi lebar selang secara sistematis sehingga lebar selang terebut semakin sempit dan karenanya menuju akar yang benar. Dalam sebuah selang mungkin terdapat lebih dari satu buah akar atau tidak ada akar sama sekali. Jumlah akar dapat diketahui dari perkalian fungsi selang dan berdasarkan grafik yang dibentuk pada Gambar 8.1 dan Gambar 8.2 berikut

a. f (a).f (b) < 0

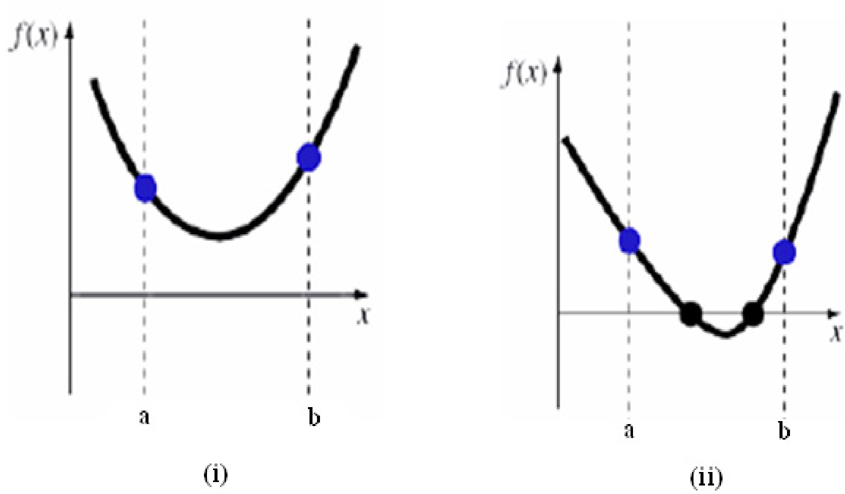
Bila f(a) dan f(b) mempunyai tanda yang berbeda, maka jumlah akar biasanya merupakan bilangan ganjil.



Gambar 8.1 f(a) dan f(b) berbeda tanda

b. f (a).f (b) >0

Bila f (a) dan f(b) mempunyai tanda yang sama, maka jumlah akar biasanya merupakan bilangan genap atau tidak ada akar sama sekali.



Gambar 8.2 f(a) dan f(b) Mempuyai Tanda Sama

Contoh 8.1: Dengan menggunakan metoda Grafis dan bantuan perangkat lunak Matlab, tentukan akar dari persamaan (8.1) berikut

$$f(x) = x^4 - 8.5000x^3 - 35.5000x^2 + 465.000x - 1000.0000 \quad \ldots\ldots\ldots\ldots \quad (8.1)$$

Jawab :

Persamaan (8.1) diselesaikan dengan menggunakan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Metode Grafis')
disp('==================')
c = input('Nilai Batas Bawah     : ');
e = input('Nilai Batas Atas      : ');
d = input('Selang Perhitungan    : ');
%
akar2=100;
flag=1;
index=1;
for x=c:d:e;
temp= (x^4) - (8.50*((x)^3)) - (35.50*x) + (465*x) - 1000;
a(index)=temp;
    if abs(temp)<abs(akar2);
        akar2=temp;
        t=x;
    end;
    y(index)=x;
index=index+1;
 disp (['Iterasi ke ',num2str(index),',',' akar :
',num2str(t)])
end;
%
disp ('  ')
disp (['Nilai Akar            : ',num2str(t)])
disp (['Jumlah Iterasi        : ',num2str(index)])
%
% Plot Grafik
plot(y,a,'-r')
text(t,akar2,['\leftarrow Akar
Persamaan',num2str(t)],'HorizontalAlignment','left')
%
```

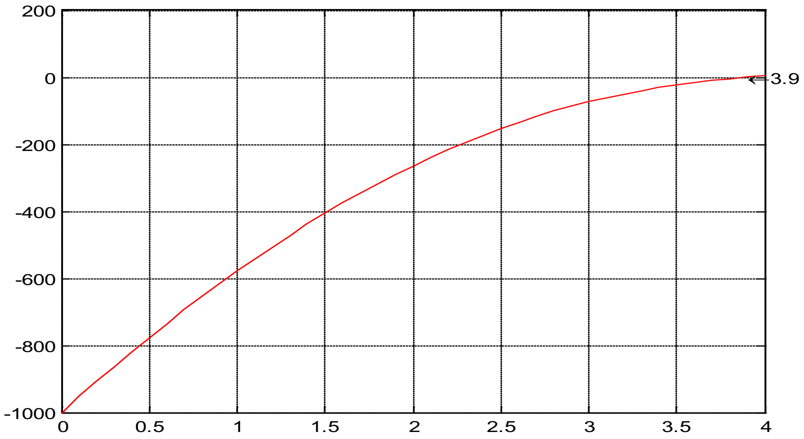
```
grid on
%end
```

Hasil program

```
Metode Grafis
===================
Nilai Batas Bawah     : 0.0000
Nilai Batas Atas      : 4.0000
Selang Perhitungan    : 0.1000
Iterasi ke 2, akar : 0
Iterasi ke 3, akar : 0.1
Iterasi ke 4, akar : 0.2
Iterasi ke 5, akar : 0.3
Iterasi ke 6, akar : 0.4
Iterasi ke 7, akar : 0.5
Iterasi ke 8, akar : 0.6
Iterasi ke 9, akar : 0.7
Iterasi ke 10, akar : 0.8
Iterasi ke 11, akar : 0.9
Iterasi ke 12, akar : 1
Iterasi ke 13, akar : 1.1
Iterasi ke 14, akar : 1.2
Iterasi ke 15, akar : 1.3
Iterasi ke 16, akar : 1.4
Iterasi ke 17, akar : 1.5
Iterasi ke 18, akar : 1.6
Iterasi ke 19, akar : 1.7
Iterasi ke 20, akar : 1.8
Iterasi ke 21, akar : 1.9
Iterasi ke 22, akar : 2
Iterasi ke 23, akar : 2.1
Iterasi ke 24, akar : 2.2
Iterasi ke 25, akar : 2.3
Iterasi ke 26, akar : 2.4
Iterasi ke 27, akar : 2.5
Iterasi ke 28, akar : 2.6
Iterasi ke 29, akar : 2.7
Iterasi ke 30, akar : 2.8
Iterasi ke 31, akar : 2.9
Iterasi ke 32, akar : 3
Iterasi ke 33, akar : 3.1
Iterasi ke 34, akar : 3.2
Iterasi ke 35, akar : 3.3
Iterasi ke 36, akar : 3.4
Iterasi ke 37, akar : 3.5
Iterasi ke 38, akar : 3.6
```

```
Iterasi ke 39, akar : 3.7
Iterasi ke 40, akar : 3.8
Iterasi ke 41, akar : 3.9
Iterasi ke 42, akar : 3.9

Nilai Akar           : 3.9
Jumlah Iterasi       : 42
```

Akar dari persamaan (8.1) adalah 3.9000 yang diperoleh pada iterasi ke 42. Adapun grafik dari persamaan (8.1) diperlihatkan pada Gambar 8.1 berikut



Gambar 8.3 Grafik Persamaan (8.1)

8.2.2. Metoda Bagi Dua

Dasar penyelesaian untuk metoda bagi dua adalah jika f (x) nyata dan kontinu dalam interval (x_l, x_u) dan f (x_l), f (x_u) berbeda tanda maka berlaku persamaan (8.2) berikut

$$f(x_a)\, f(x_b) < 0 \qquad \text{(8.2)}$$

dan sedikitnya terdapat satu akar dalam interval (x_l, x_u) tersebut. Adapun langkah – langkah penentuan akar – akar dari suatu persamaan dengan metoda bagi dua ini dinyatakan sebagai berikut

1. Tentukan harga batas bawah (x_l) dan harga batas atas (x_u) interval awal sedemikian sehingga ada perubahan tanda f (x) pada batas – batas interval tersebut atau diperiksan dengan persamaan (8.3) berikut

$$f(x_l)\, f(x_u) < 0 \quad \text{......} \quad (8.3)$$

2. Lakukan taksiran awal dari akar dengan menggunakan persamaan (8.4) berikut

$$x_r = \frac{x_l + x_u}{2} \quad \text{......} \quad (8.4)$$

3. Lakukan evaluasi untuk menentukan subinterval yang mengandung akar persamaan berikut
   a. Jika $f(x_l)\, f(x_r) < 0$, akar terletak pada subinterval pertama sehingga jadikan $x_u = x_r$ dan lanjutkan ke langkah 4.
   b. Jika $f(x_l)\, f(x_r) > 0$, akar terletak pada subinterval kedua sehingga jadikan $x_l = x_r$ dan lanjutkan ke langkah 4.
   c. Jika $f(x_l)\, f(x_r) = 0$, akar dari persamaan adalah x_r dan perhitungan selesai
4. Lakukan taksiran baru dari akar dengan menggunakan persamaan (8.5) berikut

$$x_r = \frac{x_l + x_u}{2} \quad \text{......} \quad (8.5)$$

5. Periksa kesalahan relatif (E_r) hasil perhitungan dimana
   a. Jika kesalahan relatif (E_r) besar dari tingkat ketelitian yang diizinkan maka perhitungan diulangi lagi.
   a. Jika kesalahan relatif (E_r) kecil dari tingkat ketelitian yang diizinkan maka perhitungan selesai.

dimana

$$E_r = \left| \frac{x_{r(i+1)} - x_{r(i)}}{x_{r(i+1)}} \right| .100\ \% \quad \text{......} \quad (8.6)$$

Contoh 8.2: Dengan menggunakan metoda bagi dua dan bantuan perangkat lunak Matlab tentukan akar dari persamaan (8.7) berikut

$$f(x) = x^4 - 8.5000x^3 - 35.5000x^2 + 465.000x - 1000.0000 \quad \text{......} \quad (8.7)$$

dengan tingkat ketelitian 0.001 %

Jawab :

Persamaan (3.7) diselesaikan dengan menggunakan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
%
disp('Metode Bagi Dua')
disp('===================')
a = input('Nilai Batas Bawah     : ');
b = input('Nilai Batas Atas      : ');
N = input('Jumlah Iterasi        : ');
Tol = input('Tingkat Ketelitian    : ')
syms x
f = (x^4) - (8.50*x^3) - (35.50*x^2) + (465*x) - 1000
[k,c] = bisection( f, a, b, N,Tol,1);
disp ('     ');
disp (['Nilai Akar             : ',num2str(c)])
disp (['Jumlah Iterasi         : ',num2str(k)])
```

Untuk fungsi bisection diperlihatkan dengan kode berikut

```
function [k,c] = bisection(f, a, b, N, Tol,verb)
%
if nargin == 3
        N = 1e4;
        Tol = 1e-4;
        verb = false;
    elseif nargin == 4
        Tol = 1e-4;
        verb = false;
    elseif nargin == 5
        verb = false;
    elseif nargin ~= 6
        error('bisection: invalid input parameterss');
    end

    f = inline(f);
    if (f(a) * f(b) > 0) || (a >= b)
        error('bisection: condition f(a)*f(b)>0 or a>=b didn''t 
apply');
    end
    k = 1;
```

```
    x(k) = (a + b) / 2;
    while ((k <= N) && ((b - a) / 2) >= Tol)
        if f(x(k)) == 0
            error([ 'bisection: condition f(' num2str(x(k)) ...
                ')~=0 didn''t apply' ]);
        end
        if (f(x(k)) * f(a)) < 0
            b = x(k);
        else
            a = x(k);
        end
        k = k + 1;
        x(k) = (a + b) / 2;
        if verb == true
        disp (['Iterasi ke ',num2str(k),',',' akar :
',num2str(x(k))])
        end
    end
c =  x(k);
end
```

Hasil program

```
Metode Bagi Dua
===================
Nilai Batas Bawah      : 0.0000
Nilai Batas Atas       : 5.0000
Jumlah Iterasi         : 100
Tingkat Ketelitian     : 0.00001

Tol =
   1.0000e-05
f =
x^4 - (17*x^3)/2 - (71*x^2)/2 + 465*x - 1000

Iterasi ke 2, akar : 3.75
Iterasi ke 3, akar : 4.375
Iterasi ke 4, akar : 4.0625
Iterasi ke 5, akar : 3.9063
Iterasi ke 6, akar : 3.8281
Iterasi ke 7, akar : 3.8672
Iterasi ke 8, akar : 3.8867
Iterasi ke 9, akar : 3.877
Iterasi ke 10, akar : 3.8818
Iterasi ke 11, akar : 3.8843
Iterasi ke 12, akar : 3.8831
Iterasi ke 13, akar : 3.8824
```

```
Iterasi ke 14, akar : 3.8821
Iterasi ke 15, akar : 3.8823
Iterasi ke 16, akar : 3.8824
Iterasi ke 17, akar : 3.8823
Iterasi ke 18, akar : 3.8823
Iterasi ke 19, akar : 3.8823

Nilai Akar           : 3.8823
Jumlah Iterasi       : 19
```

Akar dari persamaan (8.7) adalah 3.8823 yang diperoleh pada iterasi ke 19 dengan tingkat ketelitian 0.001 %.

8.2.3. Metoda Posisi Palsu

Pada dasarnya penentuan akar dengan menggunakan metode bagi dua selalu dapat menemukan akar, tetapi kecepatan untuk mencapai akar hampiran sangat lambat. Untuk mempercepat pencarian akar tersebut, maka f (x_1) dan f (x_u) perlu diperhitungkan. Metode yang memanfaatkan f (x_1) dan f (x_1) ini disebut dengan metode posisi palsu. Adapun langkah – langkah penentuan akar – akar dari suatu persamaan dengan metoda bagi dua ini dinyatakan sebagai berikut

1. Tentukan harga batas bawah (x_1) dan harga batas atas (x_u) interval awal sedemikian sehingga ada perubahan tanda f (x) pada batas - batas interval tersebut atau diperiksan dengan persamaan (8.8) berikut

$$f(x_1)\,f(x_u) < 0 \qquad \text{...............} \qquad (8.8)$$

2. Lakukan taksiran awal dari akar dengan menggunakan persamaan (8.9) berikut

$$x_r = x_u - \frac{f(x_u)(x_u - x_1)}{f(x_u) - f(x_1)} \qquad \text{...............} \qquad (8.9)$$

3. Lakukan evaluasi untuk menentukan subinterval yang mengandung akar persamaan berikut
   a. Jika f (x_1) f $(x_r) < 0$, akar terletak pada subinterval pertama sehingga jadikan $x_u = x_r$ dan lanjutkan ke langkah 4.
   b. Jika f (x_1) f $(x_r) > 0$, akar terletak pada subinterval kedua sehingga jadikan $x_1 = x_r$ dan lanjutkan ke langkah 4.

   c. Jika f (x_l) f (x_r) = 0, akar dari persamaan adalah x_r dan perhitungan selesai

4. Lakukan taksiran baru dari akar dengan menggunakan persamaan (8.10) berikut

$$x_r = x_u - \frac{f(x_u)(x_u - x_l)}{f(x_u) - f(x_l)} \quad \text{...} \quad (8.10)$$

5. Periksa kesalahan relatif (E_r) hasil perhitungan dimana
   a. Jika kesalahan relatif (E_r) besar dari tingkat ketelitian yang diizinkan maka perhitungan diulangi lagi.
   b. Jika kesalahan relatif (E_r) kecil dari tingkat ketelitian yang diizinkan maka perhitungan selesai. Untuk perhitungan kesalahan relatif digunakan persamaan (8.6).

Contoh 8.3: Dengan menggunakan metoda posisi palsu dan bantuan perangkat lunak Matlab tentukan akar dari persamaan (8.11) berikut

$$f(x) = x^4 - 8.5000x^3 - 35.5000x^2 + 465.000x - 1000.0000 \quad \text{.............} \quad (8.11)$$

dengan tingkat ketelitian 0.0010 %

Jawab :

Persamaan (8.11) diselesaikan dengan menggunakan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Metode Posisi Palsu')
disp('===================')
xi = input('Tebakan Batas Bawah : ');
xu = input('Tebakan Batas Atas  : ');
e  = input('Tingkat Presisi     : ');
e_temp=500;
step=1;
disp ('     ');
while e_temp>e
f_xi = (xi^4) - (8.50*xi^3) - (35.50*xi^2) + (465*xi) - 1000;
f_xu = (xu^4) - (8.50*xu^3) - (35.50*xu^2) + (465*xu) - 1000;
```

```
if (step==1)
xr=xu-(f_xu*(xi-xu)/(f_xi-f_xu));
end;
f_xr = (xr^4) - (8.50*xr^3) - (35.50*xr^2) + (465*xr) - 1000;
if (f_xi*f_xr)<0
    xu=xr;
elseif(f_xi*f_xr)>0
    xi=xr;
else
    regulafalsi=xr;
    break;
end;
f_xi = (xi^4) - (8.50*xi^3) - (35.50*xi^2) + (465*xi) - 1000;
f_xu = (xu^4) - (8.50*xu^3) - (35.50*xu^2) + (465*xu) - 1000;
xr2=xu-(f_xu*(xi-xu)/(f_xi-f_xu));
e_temp=(abs(xr2-xr)/xr2)*100;
xr=xr2;
disp (['Iterasi ke ',num2str(step),',',' akar : ',num2str(xr)])
step=step+1;
end;
disp ('     ');
disp (['Nilai Akar      : ',num2str(xr)])
disp (['Jumlah Iterasi  : ',num2str(step)])
disp (['Kesalahan       : ',num2str(e_temp)])
```

Hasil program

```
Metode Posisi Palsu
===================
Tebakan Batas Bawah : 0.0000
Tebakan Batas Atas  : 4.0000
Tingkat Presisi     : 0.00001

Iterasi ke 1, akar : 3.9701
Iterasi ke 2, akar : 3.9578
Iterasi ke 3, akar : 3.9471
Iterasi ke 4, akar : 3.9379
Iterasi ke 5, akar : 3.9299
Iterasi ke 6, akar : 3.923
Iterasi ke 7, akar : 3.917
Iterasi ke 8, akar : 3.9119
Iterasi ke 9, akar : 3.9076
Iterasi ke 10, akar : 3.9038
Iterasi ke 11, akar : 3.9006
Iterasi ke 12, akar : 3.8979
Iterasi ke 13, akar : 3.8956
Iterasi ke 14, akar : 3.8936
```

```
Iterasi ke 15, akar : 3.8919
Iterasi ke 16, akar : 3.8904
Iterasi ke 17, akar : 3.8892
Iterasi ke 18, akar : 3.8882
Iterasi ke 19, akar : 3.8873
Iterasi ke 20, akar : 3.8865
Iterasi ke 21, akar : 3.8859
Iterasi ke 22, akar : 3.8854
Iterasi ke 23, akar : 3.8849
Iterasi ke 24, akar : 3.8845
Iterasi ke 25, akar : 3.8842
Iterasi ke 26, akar : 3.8839
Iterasi ke 27, akar : 3.8836
Iterasi ke 28, akar : 3.8834
Iterasi ke 29, akar : 3.8833
Iterasi ke 30, akar : 3.8831
Iterasi ke 31, akar : 3.883
Iterasi ke 32, akar : 3.8829
Iterasi ke 33, akar : 3.8828
Iterasi ke 34, akar : 3.8827
Iterasi ke 35, akar : 3.8827
Iterasi ke 36, akar : 3.8826
Iterasi ke 37, akar : 3.8826
Iterasi ke 38, akar : 3.8825
Iterasi ke 39, akar : 3.8825
Iterasi ke 40, akar : 3.8825
Iterasi ke 41, akar : 3.8824
Iterasi ke 42, akar : 3.8824
Iterasi ke 43, akar : 3.8824
Iterasi ke 44, akar : 3.8824
Iterasi ke 45, akar : 3.8824
Iterasi ke 46, akar : 3.8824
Iterasi ke 47, akar : 3.8824
Iterasi ke 48, akar : 3.8823
Iterasi ke 49, akar : 3.8823
Iterasi ke 50, akar : 3.8823
Iterasi ke 51, akar : 3.8823
Iterasi ke 52, akar : 3.8823
Iterasi ke 53, akar : 3.8823
Iterasi ke 54, akar : 3.8823
Iterasi ke 55, akar : 3.8823
Iterasi ke 56, akar : 3.8823
Iterasi ke 57, akar : 3.8823
Iterasi ke 58, akar : 3.8823
Iterasi ke 59, akar : 3.8823
Iterasi ke 60, akar : 3.8823
Iterasi ke 61, akar : 3.8823
```

```
Iterasi ke 62, akar : 3.8823
Iterasi ke 63, akar : 3.8823
Iterasi ke 64, akar : 3.8823
Iterasi ke 65, akar : 3.8823
Iterasi ke 66, akar : 3.8823
Iterasi ke 67, akar : 3.8823

Nilai Akar       : 3.8823
Jumlah Iterasi   : 68
Kesalahan        : 8.5824e-06
```

Akar dari persamaan (8.11) adalah 3.8823 yang diperoleh pada iterasi ke 68 dengan tingkat ketelitian 0.0010 %.

8.3 Metoda Terbuka

Berbeda dengan metoda tertutup, metoda terbuka tidak memerlukan nilai batas bawah dan nilai batas atas yang mengandung akar. Pada metoda terbuka yang diperlukan adalah tebakan awal akar. Dengan prosedur iterasi, tebakan awal akar ini digunakan untuk menghitung hampiran akar yang baru. Pada setiap kali iterasi, hampiran akar yang lama dipakai untuk menghitung hampiran akar yang baru. Mungkin saja hampiran akar yang baru mendekati akar sejati (kovergen) atau mungkin juga menjauhinya (divergen) . Untuk itu metoda terbuka tidak selalu berhasil menemukan akar –akar, kadang – kadang konvergen kadangkala divergen. Pembahasan metoda terbuka akan dimulai dengan pembahasan metoda iterasi satu titik sederhana, metoda Newton Raphson dan metoda Secant.

8.3.1 Metoda Iterasi Satu Titik Sederhana

Metoda ini kadang – kadang dinamakan juga metoda iterasi sederhana, metoda langsung atau metoda substitusi beruntun. Kesederhanaan metoda ini karena pembentukan prosedur iterasinya mudah dibentuk dalam bentuk persamaan (8.12) berikut

$$x_{r+1} = g(x_r) \qquad (8.12)$$

Persamaan (8.12) diperoleh dari f(x) = 0 yang diubah menjadi bentuk persamaan (8.12). Setelah itu ditentukan tebakan awal x_0 kemudian hitung nilai x_1, x_2, x_3 dan seterusnya. Iterasi akan berhenti jika dipenuhi persamaan (8.13) berikut

$$E_r = \left| \frac{x_{r(i+1)} - x_{r(i)}}{x_{r(i+1)}} \right| .100\ \% \quad \text{......} \qquad (8.13)$$

Contoh 8.4: Dengan menggunakan metoda iterasi satu titik sederhana dan bantuan perangkat lunak Matlab, tentukan akar dari persamaan (8.14) berikut

$$f(x) = x^4 - 8.5000x^3 - 35.5000x^2 + 465.0000x - 1000.0000 \quad \text{......} \qquad (8.14)$$

dengan tingkat ketelitian 0.0010 %

Jawab :

Persamaan (8.14) diselesaikan dengan menggunakan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
[x,fval,exitflag,output]  =  fzero(@(x)  (x^4)  -  (8.50*x^3)  -
(35.50*x^2) + (465*x) - 1000 ,3)

Hasil program
x =
    3.8823

fval =
     0

exitflag =
     1

output =
    intervaliterations: 8
            iterations: 7
             funcCount: 24
             algorithm: 'bisection, interpolation'
               message: 'Zero found in the interval [2.04,
3.96]'
```

Akar dari persamaan (8.14) adalah 3.8823 yang diperoleh pada iterasi ke 7.

8.3.2 Metoda Newton Rapshon

Diantara metoda penentuan akar – akar persamaan, metoda Newton Raphson merupakan metoda yang paling banyak dipakai dalam terapan sains dan rekayasa. Metoda ini paling disukai karena konvergensinya paling cepat diantara metoda lainnya. Ada dua pendekatan yang digunakan dalam penurunan persamaan metoda Newton Raphson yaitu penurunan metoda Newton Raphson secara geometri dan penurunan metoda Newton Raphson dengan bantuan deret Taylor. Adapun langkah – langkah penentuan akar dari suatu persamaan dengan metoda Newton Raphson ini dinyatakan sebagai berikut

1. Tentukan taksiran awal akar untuk fungsi f(x). Untuk taksiran awal dari akar fungsi f(x) dinyatakan dalam bentuk x_a
2. Tentukan turunan pertama dari fungsi f(x). Untuk turunan pertama dari f(x) dinyatakan dalam bentuk $f'(x)$
3. Lakukan evaluasi f(x) dan $f'(x)$ untuk $x = x_a$
4. Hitung pendekatan akar yang baru dari fungsi f(x) dengan menggunakan persamaan (8.15) berikut

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)} \quad \text{dimana} \quad f'(x_r) \neq 0 \quad \dots\dots\dots \quad (8.15)$$

5. Periksa kesalahan relatif (E_r) hasil perhitungan dimana
   a. Jika kesalahan relatif (E_r) besar dari tingkat ketelitian yang diizinkan maka perhitungan diulangi lagi langkah ke 2 sampai langkah ke 4.
   b. Jika kesalahan relatif (E_r) kecil dari tingkat ketelitian yang diizinkan maka perhitungan selesai. Untuk perhitungan kesalahan relatif digunakan persamaan (8.6).

Contoh 8.5: Dengan menggunakan metoda Newton Raphson dan bantuan perangkat lunak Matlab, tentukan akar dari persamaan (8.16) berikut

$$f(x) = x^4 - 8.5000x^3 - 35.5000x^2 + 465.0000x - 1000.0000 \quad \dots\dots\dots \quad (8.16)$$

dengan tingkat ketelitian 0.0010 %

Jawab :

Persamaan (8.16) diselesaikan dengan menggunakan Matlab dengan kode sebagai berikut

```
function newton_raphson ;
clc
clear all
close all
%
disp('Metoda Newton Raphson');
disp('=====================');
x =    input('Nilai Tebakan Awal        : ');
eps =  input('Nilai Ketelitian          : ');
itemax = input('Jumlah Iterasi Maksimum   : ');
alfa = x;
disp ('     ');
disp ('Proses Iterasi')
disp ('=====================')
xlama = x;
tic
for i = 1:1:itemax
   if ft(xlama) == 0
      break;
   else
      xbaru = xlama - (fn(xlama,alfa)/ft(xlama));
   end
   disp (['Iterasi ke ',num2str(i),',',' akar :
',num2str(xbaru)])
   if abs((xbaru-xlama)/xbaru) <= eps
      break;
   end
   xlama = xbaru;
end
salah = abs((xbaru-xlama)/xbaru);
disp ('  ')
disp (['Nilai Akar      : ',num2str(xbaru)])
disp (['Jumlah Iterasi  : ',num2str(i)])
disp (['Kesalahan       : ',num2str(salah)])

function f = fn(x,alfa);
f = x^4 - 8.5000*x^3 - 35.5000*x^2 + 465.0000*x - 1000.0000;


function f = ft(x);
f = 4*x^3 - 25.5000*x^2 - 71.0000*x + 465.0000;
```

Hasil program

```
Metoda Newton Raphson
======================
Nilai Tebakan Awal        : 3.0000
Nilai Ketelitian          : 0.00001
Jumlah Iterasi Maksimum   : 100
Proses Iterasi
======================
Iterasi ke 1, akar : 3.5594
Iterasi ke 2, akar : 3.81
Iterasi ke 3, akar : 3.8771
Iterasi ke 4, akar : 3.8823
Iterasi ke 5, akar : 3.8823

Nilai Akar       : 3.8823
Jumlah Iterasi   : 5
Kesalahan        : 7.7129e-06
```

Akar dari persamaan (8.16) adalah 3.8823 yang diperoleh pada iterasi ke 5 dengan tingkat ketelitian kecil dari 0.0010 %.

8.3.3 Metoda Secant

Pada dasarnya metoda Secant ini sama dengan metoda Newton-Raphson, perbedaannya hanya terletak pada pendekatan untuk turunan pertama dari f(x) saja. Selain itu pada dasarnya metoda Secant ini identik dengan metoda posisi palsu. Perbedaannya adalah metoda posisi palsu selalu menggantikan salah satu dari dua taksiran akar sehingga akar selalu dalam keadaan terkurung dan titik-titik lama selalu diperbaharui menjadi titik yang baru, sedangkan metoda Secant tidak memerlukan dua taksiran awal yang harus mengurung akar persamaan. Adapun langkah – langkah penentuan akar dari suatu persamaan dengan metoda Secant ini dinyatakan sebagai berikut

1. Tentukan 2 taksiran awal akar untuk fungsi f(x). Untuk 2 taksiran awal dari akar fungsi f(x) dinyatakan dalam bentuk x_0 dan x_1
2. Lakukan evaluasi f(x) untuk taksiran awal akar x_0 dan x_1 serta diperoleh $f(x_0)$ dan $f(x_1)$.
3. Hitung pendekatan akar yang baru dari fungsi f(x) dengan menggunakan persamaan (8.17) berikut

$$x_{r+1} = x_r - \left(\frac{f(x_r)(x_r - x_{r-1})}{f(x_r) - f(x_{r-1})} \right) \quad \text{...} \quad (8.17)$$

3. Periksa kesalahan relatif (E_r) hasil perhitungan dimana
   a. Jika kesalahan relatif (E_r) besar dari tingkat ketelitian yang diizinkan maka perhitungan diulangi lagi dengan $x_0 = x_r$ dan $x_1 = x_{r+1}$
   b. Jika kesalahan relatif (E_r) kecil dari tingkat ketelitian yang diizinkan maka perhitungan selesai. Untuk perhitungan kesalahan relatif digunakan persamaan (8.6).

Contoh 8.6: Dengan menggunakan metoda Secant dan bantuan perangkat lunak Matlab, tentukan akar dari persamaan (8.18) berikut

$$f(x) = x^4 - 8.5000x^3 - 35.5000x^2 + 465.0000x - 1000.0000 \quad \text{............} \quad (8.18)$$

dengan tingkat ketelitian 0.0010 %

Jawab :

Persamaan (8.18) diselesaikan dengan menggunakan Matlab dengan kode sebagai

```
clc
clear all
close all
close all hidden
%
disp('Metoda Secant');
disp('=================================');
x0  = input('Masukkan x0                  : ');
x1  = input('Masukkan x1                  : ');
E   = input('Nilai Ketelitian             : ');
disp ('     ');
i=0;
M = 9;
while (E<M)
   fx0= (x0^4) - (8.5000*(x0^3)) - (35.5000*x0^2) +
(465.0000*x0) - 1000.0000;
   fx1= (x0^4) - (8.5000*(x1^3)) - (35.5000*x1^2) +
(465.0000*x1) - 1000.0000;
   d = x1 - (fx1*(x1-x0)/(fx1-fx0));
   M=abs(x1-x0);
```

```
   x0 = x1;
   x1 = d;
   i=i+1;
   disp (['Iterasi ke ',num2str(i),',',' akar : ',num2str(x1)])
end;
disp ('  ')
disp (['Nilai Akar      : ',num2str(x1)])
disp (['Jumlah Iterasi  : ',num2str(i)])
disp (['Kesalahan       : ',num2str(M)])
```

Hasil program

```
Metoda Secant
=================================
Masukkan x0             : 3.0000
Masukkan x1             : 4.0000
Nilai Ketelitian        : 0.00001

Iterasi ke 1, akar : 2.2551
Iterasi ke 2, akar : 4.3087
Iterasi ke 3, akar : -1.6864
Iterasi ke 4, akar : 4.2721
Iterasi ke 5, akar : 5.5107
Iterasi ke 6, akar : 4.2902
Iterasi ke 7, akar : 5.4789
Iterasi ke 8, akar : 4.3087
Iterasi ke 9, akar : 5.449
Iterasi ke 10, akar : 4.3274
Iterasi ke 11, akar : 5.4209
Iterasi ke 12, akar : 4.3465
Iterasi ke 13, akar : 5.3945
Iterasi ke 14, akar : 4.3657
Iterasi ke 15, akar : 5.3698
Iterasi ke 16, akar : 4.3851
Iterasi ke 17, akar : 5.3466
Iterasi ke 18, akar : 4.4046
Iterasi ke 19, akar : 5.325
Iterasi ke 20, akar : 4.4241
Iterasi ke 21, akar : 5.3047
Iterasi ke 22, akar : 4.4435
Iterasi ke 23, akar : 5.2858
Iterasi ke 24, akar : 4.4629
Iterasi ke 25, akar : 5.2681
Iterasi ke 26, akar : 4.4821
Iterasi ke 27, akar : 5.2515
Iterasi ke 28, akar : 4.5011
Iterasi ke 29, akar : 5.2361
```

```
Iterasi ke 30, akar : 4.5198
Iterasi ke 31, akar : 5.2216
Iterasi ke 32, akar : 4.5383
Iterasi ke 33, akar : 5.2081
Iterasi ke 34, akar : 4.5564
Iterasi ke 35, akar : 5.1955
Iterasi ke 36, akar : 4.5742
Iterasi ke 37, akar : 5.1838
Iterasi ke 38, akar : 4.5915
Iterasi ke 39, akar : 5.1727
Iterasi ke 40, akar : 4.6085
Iterasi ke 41, akar : 5.1624
Iterasi ke 42, akar : 4.625
Iterasi ke 43, akar : 5.1528
Iterasi ke 44, akar : 4.641
Iterasi ke 45, akar : 5.1438
Iterasi ke 46, akar : 4.6566
Iterasi ke 47, akar : 5.1353
Iterasi ke 48, akar : 4.6717
Iterasi ke 49, akar : 5.1274
Iterasi ke 50, akar : 4.6863
Iterasi ke 51, akar : 5.12
Iterasi ke 52, akar : 4.7004
Iterasi ke 53, akar : 5.113
Iterasi ke 54, akar : 4.714
Iterasi ke 55, akar : 5.1065
Iterasi ke 56, akar : 4.7271
Iterasi ke 57, akar : 5.1004
Iterasi ke 58, akar : 4.7397
Iterasi ke 59, akar : 5.0947
Iterasi ke 60, akar : 4.7519
Iterasi ke 61, akar : 5.0893
Iterasi ke 62, akar : 4.7635
Iterasi ke 63, akar : 5.0842
Iterasi ke 64, akar : 4.7747
Iterasi ke 65, akar : 5.0795
Iterasi ke 66, akar : 4.7854
Iterasi ke 67, akar : 5.075
Iterasi ke 68, akar : 4.7957
Iterasi ke 69, akar : 5.0708
Iterasi ke 70, akar : 4.8056
Iterasi ke 71, akar : 5.0668
Iterasi ke 72, akar : 4.815
Iterasi ke 73, akar : 5.0631
Iterasi ke 74, akar : 4.824
Iterasi ke 75, akar : 5.0596
Iterasi ke 76, akar : 4.8326
```

```
Iterasi ke 77, akar : 5.0563
Iterasi ke 78, akar : 4.8408
Iterasi ke 79, akar : 5.0532
Iterasi ke 80, akar : 4.8486
Iterasi ke 81, akar : 5.0503
Iterasi ke 82, akar : 4.8561
Iterasi ke 83, akar : 5.0475
Iterasi ke 84, akar : 4.8633
Iterasi ke 85, akar : 5.0449
Iterasi ke 86, akar : 4.8701
Iterasi ke 87, akar : 5.0425
Iterasi ke 88, akar : 4.8765
Iterasi ke 89, akar : 5.0401
Iterasi ke 90, akar : 4.8827
Iterasi ke 91, akar : 5.038
Iterasi ke 92, akar : 4.8886
Iterasi ke 93, akar : 5.0359
Iterasi ke 94, akar : 4.8942
Iterasi ke 95, akar : 5.034
Iterasi ke 96, akar : 4.8995
Iterasi ke 97, akar : 5.0321
Iterasi ke 98, akar : 4.9046
Iterasi ke 99, akar : 5.0304
Iterasi ke 100, akar : 4.9094
Iterasi ke 101, akar : 5.0288
Iterasi ke 102, akar : 4.914
Iterasi ke 103, akar : 5.0272
Iterasi ke 104, akar : 4.9184
Iterasi ke 105, akar : 5.0257
Iterasi ke 106, akar : 4.9225
Iterasi ke 107, akar : 5.0244
Iterasi ke 108, akar : 4.9265
Iterasi ke 109, akar : 5.0231
Iterasi ke 110, akar : 4.9302
Iterasi ke 111, akar : 5.0218
Iterasi ke 112, akar : 4.9338
Iterasi ke 113, akar : 5.0207
Iterasi ke 114, akar : 4.9372
Iterasi ke 115, akar : 5.0196
Iterasi ke 116, akar : 4.9404
Iterasi ke 117, akar : 5.0185
Iterasi ke 118, akar : 4.9435
Iterasi ke 119, akar : 5.0175
Iterasi ke 120, akar : 4.9464
Iterasi ke 121, akar : 5.0166
Iterasi ke 122, akar : 4.9491
Iterasi ke 123, akar : 5.0157
```

```
Iterasi ke 124, akar : 4.9517
Iterasi ke 125, akar : 5.0149
Iterasi ke 126, akar : 4.9542
Iterasi ke 127, akar : 5.0141
Iterasi ke 128, akar : 4.9566
Iterasi ke 129, akar : 5.0134
Iterasi ke 130, akar : 4.9588
Iterasi ke 131, akar : 5.0126
Iterasi ke 132, akar : 4.9609
Iterasi ke 133, akar : 5.012
Iterasi ke 134, akar : 4.9629
Iterasi ke 135, akar : 5.0113
Iterasi ke 136, akar : 4.9648
Iterasi ke 137, akar : 5.0107
Iterasi ke 138, akar : 4.9667
Iterasi ke 139, akar : 5.0102
Iterasi ke 140, akar : 4.9684
Iterasi ke 141, akar : 5.0096
Iterasi ke 142, akar : 4.97
Iterasi ke 143, akar : 5.0091
Iterasi ke 144, akar : 4.9716
Iterasi ke 145, akar : 5.0087
Iterasi ke 146, akar : 4.973
Iterasi ke 147, akar : 5.0082
Iterasi ke 148, akar : 4.9744
Iterasi ke 149, akar : 5.0078
Iterasi ke 150, akar : 4.9758
Iterasi ke 151, akar : 5.0074
Iterasi ke 152, akar : 4.977
Iterasi ke 153, akar : 5.007
Iterasi ke 154, akar : 4.9782
Iterasi ke 155, akar : 5.0066
Iterasi ke 156, akar : 4.9793
Iterasi ke 157, akar : 5.0063
Iterasi ke 158, akar : 4.9804
Iterasi ke 159, akar : 5.0059
Iterasi ke 160, akar : 4.9814
Iterasi ke 161, akar : 5.0056
Iterasi ke 162, akar : 4.9824
Iterasi ke 163, akar : 5.0053
Iterasi ke 164, akar : 4.9833
Iterasi ke 165, akar : 5.005
Iterasi ke 166, akar : 4.9842
Iterasi ke 167, akar : 5.0048
Iterasi ke 168, akar : 4.985
Iterasi ke 169, akar : 5.0045
Iterasi ke 170, akar : 4.9858
```

```
Iterasi ke 171, akar : 5.0043
Iterasi ke 172, akar : 4.9865
Iterasi ke 173, akar : 5.0041
Iterasi ke 174, akar : 4.9872
Iterasi ke 175, akar : 5.0039
Iterasi ke 176, akar : 4.9879
Iterasi ke 177, akar : 5.0037
Iterasi ke 178, akar : 4.9885
Iterasi ke 179, akar : 5.0035
Iterasi ke 180, akar : 4.9891
Iterasi ke 181, akar : 5.0033
Iterasi ke 182, akar : 4.9897
Iterasi ke 183, akar : 5.0031
Iterasi ke 184, akar : 4.9902
Iterasi ke 185, akar : 5.0029
Iterasi ke 186, akar : 4.9907
Iterasi ke 187, akar : 5.0028
Iterasi ke 188, akar : 4.9912
Iterasi ke 189, akar : 5.0026
Iterasi ke 190, akar : 4.9917
Iterasi ke 191, akar : 5.0025
Iterasi ke 192, akar : 4.9921
Iterasi ke 193, akar : 5.0024
Iterasi ke 194, akar : 4.9925
Iterasi ke 195, akar : 5.0023
Iterasi ke 196, akar : 4.9929
Iterasi ke 197, akar : 5.0021
Iterasi ke 198, akar : 4.9933
Iterasi ke 199, akar : 5.002
Iterasi ke 200, akar : 4.9936
Iterasi ke 201, akar : 5.0019
Iterasi ke 202, akar : 4.9939
Iterasi ke 203, akar : 5.0018
Iterasi ke 204, akar : 4.9943
Iterasi ke 205, akar : 5.0017
Iterasi ke 206, akar : 4.9946
Iterasi ke 207, akar : 5.0016
Iterasi ke 208, akar : 4.9948
Iterasi ke 209, akar : 5.0015
Iterasi ke 210, akar : 4.9951
Iterasi ke 211, akar : 5.0015
Iterasi ke 212, akar : 4.9954
Iterasi ke 213, akar : 5.0014
Iterasi ke 214, akar : 4.9956
Iterasi ke 215, akar : 5.0013
Iterasi ke 216, akar : 4.9958
Iterasi ke 217, akar : 5.0012
```

```
Iterasi ke 218, akar : 4.9961
Iterasi ke 219, akar : 5.0012
Iterasi ke 220, akar : 4.9963
Iterasi ke 221, akar : 5.0011
Iterasi ke 222, akar : 4.9965
Iterasi ke 223, akar : 5.0011
Iterasi ke 224, akar : 4.9966
Iterasi ke 225, akar : 5.001
Iterasi ke 226, akar : 4.9968
Iterasi ke 227, akar : 5.001
Iterasi ke 228, akar : 4.997
Iterasi ke 229, akar : 5.0009
Iterasi ke 230, akar : 4.9971
Iterasi ke 231, akar : 5.0009
Iterasi ke 232, akar : 4.9973
Iterasi ke 233, akar : 5.0008
Iterasi ke 234, akar : 4.9974
Iterasi ke 235, akar : 5.0008
Iterasi ke 236, akar : 4.9976
Iterasi ke 237, akar : 5.0007
Iterasi ke 238, akar : 4.9977
Iterasi ke 239, akar : 5.0007
Iterasi ke 240, akar : 4.9978
Iterasi ke 241, akar : 5.0007
Iterasi ke 242, akar : 4.9979
Iterasi ke 243, akar : 5.0006
Iterasi ke 244, akar : 4.998
Iterasi ke 245, akar : 5.0006
Iterasi ke 246, akar : 4.9981
Iterasi ke 247, akar : 5.0006
Iterasi ke 248, akar : 4.9982
Iterasi ke 249, akar : 5.0005
Iterasi ke 250, akar : 4.9983
Iterasi ke 251, akar : 5.0005
Iterasi ke 252, akar : 4.9984
Iterasi ke 253, akar : 5.0005
Iterasi ke 254, akar : 4.9985
Iterasi ke 255, akar : 5.0005
Iterasi ke 256, akar : 4.9986
Iterasi ke 257, akar : 5.0004
Iterasi ke 258, akar : 4.9986
Iterasi ke 259, akar : 5.0004
Iterasi ke 260, akar : 4.9987
Iterasi ke 261, akar : 5.0004
Iterasi ke 262, akar : 4.9988
Iterasi ke 263, akar : 5.0004
Iterasi ke 264, akar : 4.9988
```

```
Iterasi ke 265, akar : 5.0003
Iterasi ke 266, akar : 4.9989
Iterasi ke 267, akar : 5.0003
Iterasi ke 268, akar : 4.999
Iterasi ke 269, akar : 5.0003
Iterasi ke 270, akar : 4.999
Iterasi ke 271, akar : 5.0003
Iterasi ke 272, akar : 4.9991
Iterasi ke 273, akar : 5.0003
Iterasi ke 274, akar : 4.9991
Iterasi ke 275, akar : 5.0003
Iterasi ke 276, akar : 4.9992
Iterasi ke 277, akar : 5.0003
Iterasi ke 278, akar : 4.9992
Iterasi ke 279, akar : 5.0002
Iterasi ke 280, akar : 4.9992
Iterasi ke 281, akar : 5.0002
Iterasi ke 282, akar : 4.9993
Iterasi ke 283, akar : 5.0002
Iterasi ke 284, akar : 4.9993
Iterasi ke 285, akar : 5.0002
Iterasi ke 286, akar : 4.9994
Iterasi ke 287, akar : 5.0002
Iterasi ke 288, akar : 4.9994
Iterasi ke 289, akar : 5.0002
Iterasi ke 290, akar : 4.9994
Iterasi ke 291, akar : 5.0002
Iterasi ke 292, akar : 4.9995
Iterasi ke 293, akar : 5.0002
Iterasi ke 294, akar : 4.9995
Iterasi ke 295, akar : 5.0002
Iterasi ke 296, akar : 4.9995
Iterasi ke 297, akar : 5.0001
Iterasi ke 298, akar : 4.9995
Iterasi ke 299, akar : 5.0001
Iterasi ke 300, akar : 4.9996
Iterasi ke 301, akar : 5.0001
Iterasi ke 302, akar : 4.9996
Iterasi ke 303, akar : 5.0001
Iterasi ke 304, akar : 4.9996
Iterasi ke 305, akar : 5.0001
Iterasi ke 306, akar : 4.9996
Iterasi ke 307, akar : 5.0001
Iterasi ke 308, akar : 4.9996
Iterasi ke 309, akar : 5.0001
Iterasi ke 310, akar : 4.9997
Iterasi ke 311, akar : 5.0001
```

```
Iterasi ke 312, akar : 4.9997
Iterasi ke 313, akar : 5.0001
Iterasi ke 314, akar : 4.9997
Iterasi ke 315, akar : 5.0001
Iterasi ke 316, akar : 4.9997
Iterasi ke 317, akar : 5.0001
Iterasi ke 318, akar : 4.9997
Iterasi ke 319, akar : 5.0001
Iterasi ke 320, akar : 4.9997
Iterasi ke 321, akar : 5.0001
Iterasi ke 322, akar : 4.9998
Iterasi ke 323, akar : 5.0001
Iterasi ke 324, akar : 4.9998
Iterasi ke 325, akar : 5.0001
Iterasi ke 326, akar : 4.9998
Iterasi ke 327, akar : 5.0001
Iterasi ke 328, akar : 4.9998
Iterasi ke 329, akar : 5.0001
Iterasi ke 330, akar : 4.9998
Iterasi ke 331, akar : 5.0001
Iterasi ke 332, akar : 4.9998
Iterasi ke 333, akar : 5.0001
Iterasi ke 334, akar : 4.9998
Iterasi ke 335, akar : 5.0001
Iterasi ke 336, akar : 4.9998
Iterasi ke 337, akar : 5.0001
Iterasi ke 338, akar : 4.9998
Iterasi ke 339, akar : 5
Iterasi ke 340, akar : 4.9998
Iterasi ke 341, akar : 5
Iterasi ke 342, akar : 4.9999
Iterasi ke 343, akar : 5
Iterasi ke 344, akar : 4.9999
Iterasi ke 345, akar : 5
Iterasi ke 346, akar : 4.9999
Iterasi ke 347, akar : 5
Iterasi ke 348, akar : 4.9999
Iterasi ke 349, akar : 5
Iterasi ke 350, akar : 4.9999
Iterasi ke 351, akar : 5
Iterasi ke 352, akar : 4.9999
Iterasi ke 353, akar : 5
Iterasi ke 354, akar : 4.9999
Iterasi ke 355, akar : 5
Iterasi ke 356, akar : 4.9999
Iterasi ke 357, akar : 5
Iterasi ke 358, akar : 4.9999
```

```
Iterasi ke 359, akar : 5
Iterasi ke 360, akar : 4.9999
Iterasi ke 361, akar : 5
Iterasi ke 362, akar : 4.9999
Iterasi ke 363, akar : 5
Iterasi ke 364, akar : 4.9999
Iterasi ke 365, akar : 5
Iterasi ke 366, akar : 4.9999
Iterasi ke 367, akar : 5
Iterasi ke 368, akar : 4.9999
Iterasi ke 369, akar : 5
Iterasi ke 370, akar : 4.9999
Iterasi ke 371, akar : 5
Iterasi ke 372, akar : 4.9999
Iterasi ke 373, akar : 5
Iterasi ke 374, akar : 4.9999
Iterasi ke 375, akar : 5
Iterasi ke 376, akar : 4.9999
Iterasi ke 377, akar : 5
Iterasi ke 378, akar : 4.9999
Iterasi ke 379, akar : 5
Iterasi ke 380, akar : 4.9999
Iterasi ke 381, akar : 5
Iterasi ke 382, akar : 5
Iterasi ke 383, akar : 5
Iterasi ke 384, akar : 5
Iterasi ke 385, akar : 5
Iterasi ke 386, akar : 5
Iterasi ke 387, akar : 5
Iterasi ke 388, akar : 5
Iterasi ke 389, akar : 5
Iterasi ke 390, akar : 5
Iterasi ke 391, akar : 5
Iterasi ke 392, akar : 5
Iterasi ke 393, akar : 5
Iterasi ke 394, akar : 5
Iterasi ke 395, akar : 5
Iterasi ke 396, akar : 5
Iterasi ke 397, akar : 5
Iterasi ke 398, akar : 5
Iterasi ke 399, akar : 5
Iterasi ke 400, akar : 5
Iterasi ke 401, akar : 5
Iterasi ke 402, akar : 5
Iterasi ke 403, akar : 5
Iterasi ke 404, akar : 5
Iterasi ke 405, akar : 5
```

```
Iterasi ke 406, akar : 5
Iterasi ke 407, akar : 5
Iterasi ke 408, akar : 5
Iterasi ke 409, akar : 5
Iterasi ke 410, akar : 5
Iterasi ke 411, akar : 5
Iterasi ke 412, akar : 5
Iterasi ke 413, akar : 5
Iterasi ke 414, akar : 5
Iterasi ke 415, akar : 5
Iterasi ke 416, akar : 5
Iterasi ke 417, akar : 5
Iterasi ke 418, akar : 5
Iterasi ke 419, akar : 5
Iterasi ke 420, akar : 5
Iterasi ke 421, akar : 5
Iterasi ke 422, akar : 5
Iterasi ke 423, akar : 5
Iterasi ke 424, akar : 5
Iterasi ke 425, akar : 5
Iterasi ke 426, akar : 5
Iterasi ke 427, akar : 5
Iterasi ke 428, akar : 5
Iterasi ke 429, akar : 5
Iterasi ke 430, akar : 5
Iterasi ke 431, akar : 5
Iterasi ke 432, akar : 5
Iterasi ke 433, akar : 5
Iterasi ke 434, akar : 5
Iterasi ke 435, akar : 5
Iterasi ke 436, akar : 5
Iterasi ke 437, akar : 5
Iterasi ke 438, akar : 5
Iterasi ke 439, akar : 5
Iterasi ke 440, akar : 5
Iterasi ke 441, akar : 5
Iterasi ke 442, akar : 5
Iterasi ke 443, akar : 5
Iterasi ke 444, akar : 5
Iterasi ke 445, akar : 5
Iterasi ke 446, akar : 5
Iterasi ke 447, akar : 5
Iterasi ke 448, akar : 5
Iterasi ke 449, akar : 5
Iterasi ke 450, akar : 5
Iterasi ke 451, akar : 5
Iterasi ke 452, akar : 5
```

```
Iterasi ke 453, akar : 5

Nilai Akar       : 5
Jumlah Iterasi   : 453
Kesalahan        : 9.894e-06
```

Akar dari persamaan (8.18) adalah 5.0000 yang diperoleh pada iterasi ke 453 dengan tingkat ketelitian kecil dari 0.001 %.

8.4 Rangkuman

Untuk penentuan akar – akar persamaan dari fungsi yang bersifat tidak linear dilakukan dengan menggunakan metoda tertutup dan metoda terbuka. Untuk metoda tertutup terdiri dari metoda grafis, metoda bagi dua dan metoda posisi palsu. Untuk metoda terbuka terdiri dari metoda iterasi satu titik sederhana, metoda Newton Raphson dan metoda Secant. Metoda tertutup ini memerlukan tebakan akar untuk batas atas dan tebakan akar untuk batas bawah. Kelebihan dari metoda tertutup ini selalu bersifat konvergen karena dalam selang batas atas dan batas bawah sudah dipastikan berisi minimal satu buah akar, sedangkan kekurangan dari metoda tertutup ini proses iterasinya lebih banyak yang mengakibatkan waktu yang diperlukan menjadi lebih lama. Untuk metoda terbuka tidak memerlukan tebakan akar untuk batas atas dan tebakan akar untuk batas bawah. Proses mencari akan dimulai dari sembarang nilai tebakan awal selanjutnya diperbaiki untuk menghitung tebakan akar yang baru. Jumlah iterasi dan waktu yang dibutuhkan dalam proses iterasi sangat tergantung dari nilai tebakan awal dan ada kemungkinan akar yang baru menjauhi dari nilai sebenarnya. Kelebihan dari metode ini lebih sedikit iterasi dan singkat waktu hitungya.

8.5 Soal - Soal

Soal 8.1: Dengan menggunakan Matlab, tentukan akar – akar dari persamaan (8.19) berikut

$$f(x) = \frac{1 - 0.6000x}{x} \quad \text{...} \quad (8.19)$$

dengan menggunakan

a. Metoda grafik.

b. Metoda bagi dua dengan tebakan awal $x_l = 1.2000$ dan $x_u = 2.2000$ serta tingkat ketelitian 0.0010

c. Metoda posisi palsu dengan tebakan awal $x_l = 1.5000$ dan $x_u = 2.0000$ serta tingkat ketelitian 0.0010
d. Metoda iterasi satu titik sederhana dengn tebakan awal $x_i = 1.4000$ serta tingkat ketelitian 0.0010
e. Metoda Newton Raphson dengan tebakan awal $x_i = 1.2500$ serta tingkat ketelitian 0.0010
f. Metoda Secant dengan tebakan awal $x_{i-1} = 1.4500$ dan $x_i = 2.2500$ serta tingkat ketelitian 0.0010

Soal 8.2: Dengan menggunakan Matlab, tentukan akar – akar dari persamaan (8.20) berikut

$$f(x) = -0.8740x^2 + 1.7500x + 2.6270 \quad \text{...} \quad (8.20)$$

dengan menggunakan

a. Metoda grafik.
b. Metoda bagi dua dengan tebakan awal $x_l = 2.9000$ dan $x_u = 3.1000$ serta tingkat ketelitian 0.0010
c. Metoda posisi palsu dengan tebakan awal $x_l = 2.7500$ dan $x_u = 3.5000$ serta tingkat ketelitian 0.0010
d. Metoda iterasi satu titik sederhana dengn tebakan awal $x_l = 2.3800$ serta tingkat ketelitian 0.0010
e. Metoda Newton Raphson dengan tebakan awal $x_l = 4.0000$ serta tingkat ketelitian 0.0010
f. Metoda Secant dengan tebakan awal $x_{l-1} = 4.1000$ dan $x_l = 4.5000$ serta tingkat ketelitian 0.0010

BAB IX
SISTEM PERSAMAAN LINIER

9.1 Pendahuluan

Pada bagian ini dibahas pengertian sistem persamaan linier, jenis – jenis matrik yang sering ditemui dalam sistem persamaan linier dan operasi baris elementer.

9.1.1. Pengertian Sistem Persamaan Linier

Suatu sistem sembarang yang terdiri dari n persamaan linier dengan n bilangan anu disebut sistem persamaan linier. Bentuk umum dari sistem persamaan linier dinyatakan dalam bentuk persamaan (9.1) berikut

$$\left.\begin{array}{c} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = c_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = c_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n = c_n \end{array}\right\} \quad \ldots\ldots\ldots\ldots \quad (9.1)$$

Persamaan (9.1) dapat juga dituliskan dalam bentuk notasi matriks yang dinyatakan dalam bentuk persamaan (9.2) dan (9.3) berikut

$$Ax = C \quad \ldots\ldots\ldots\ldots \quad (9.2)$$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_n \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.3)$$

dengan membandingkan persamaan (9.2) dan (9.3) diperoleh persamaan (9.4), (9.5) dan (9.6) berikut

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.4)$$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} \quad \text{...} \quad (9.5)$$

$$C = \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_n \end{bmatrix} \quad \text{...} \quad (9.6)$$

dimana A adalah matrik koefesien sistem persamaan linier, x adalah vektor bilangan anu dan C adalah vektor konstanta. Selain itu dalam analisa sistem persamaan linier dengan n persamaan dan n bilangan anu dinyatakan dengan suatu matrik ukuran n x (n + 1) dalam bentuk persamaan (9.7) berikut

$$AC = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & c_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & c_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n1} & \cdots & a_{nn} & c_n \end{bmatrix} \quad \text{..} \quad (9.7)$$

Susunan matrik pada persamaan (4.7) disebut matrik yang dilengkapi.

Contoh 9.1: Tentukan matrik koefesien sistem persamaan linear, vektor bilangan anu dan matrik yang dilengkapi dari sistem persamaan linier pada persamaan (9.8) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \quad \text{..................................} \quad (9.8) \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned}$$

Jawab :

Sistem persamaan linier pada persamaan (9.8) merupakan sistem persamaan linier dengan 3 persamaan dan 3 bilangan anu. Bentuk persamaan (9.8) dapat diubah menjadi persamaan (9.9) berikut

$$\begin{bmatrix} 1.0000 & 1.0000 & 2.0000 \\ 2.0000 & 4.0000 & -3.0000 \\ 3.0000 & 6.0000 & -5.0000 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.0000 \\ 1.0000 \\ 0.0000 \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.9)$$

Berdasarkan persamaan (9.9) diperoleh matrik koefesien sistem persamaan linier yang dinyatakan dalam bentuk persamaan (9.10) berikut

$$A = \begin{bmatrix} 1.0000 & 1.0000 & 2.0000 \\ 2.0000 & 4.0000 & -3.0000 \\ 3.0000 & 6.0000 & -5.0000 \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.10)$$

Vektor bilangan anu dinyatakan dalam bentuk persamaan (9.11) berikut

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.11)$$

Vektor konstanta dinyatakan dalam bentuk persamaan (9.12) berikut

$$C = \begin{bmatrix} 9.0000 \\ 1.0000 \\ 0.0000 \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.12)$$

9.1.2. Jenis – Jenis Matrik

Ada beberapa jenis matrik yang perlu diketahui dalam analisa sistem persamaan linier diantaranya

a. Matrik Bujursangkar

Matrik bujursangkar adalah matrik yang mempuyai jumlah baris dan jumlah kolom.

Contoh 9.2: Suatu matrik bujursangkar dengan dimensi 3 x 3 dinyatakan dalam bentuk persamaan (9.13) berikut

$$A = \begin{bmatrix} 1.0000 & 1.0000 & 2.0000 \\ 2.0000 & 4.0000 & 3.0000 \\ 3.0000 & 6.0000 & 5.0000 \end{bmatrix} \quad \ldots\ldots\ldots\ldots \quad (9.13)$$

Matrik pada persamaan (9.13) direpresentasikan dengan menggunakan Matlab dengan kode berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000; 2.0000  4.0000  3.0000; 3.0000
6.0000  5.0000]
disp('Dimensi Matrik A')
T = size(A)
```

Hasil program

```
Matrik A
A =
     1     1     2
     2     4     3
     3     6     5

Dimensi Matrik A
T =
     3     3
```

b. Matrik Simetri

Matrik simetri adalah matrik yang elemen a_{ij} sama dengan elemen a_{ij} untuk semua i dan j dimana dinyatakan dalam bentu persamaan (9.14) berikut

$$a_{ij} = a_{ji} \quad \text{...} \quad (9.14)$$

Contoh 9.3: Suatu matrik bujursangkar dan simetri dengan dimensi 3 x 3 dinyatakan dalam bentuk persamaan (9.15) berikut

$$A=\begin{bmatrix} 5.0000 & 1.0000 & 3.0000 \\ 1.0000 & 4.0000 & 6.0000 \\ 3.0000 & 6.0000 & 8.0000 \end{bmatrix} \quad \text{..} \quad (9.15)$$

Matrik pada persamaan (9.15) direpresentasikan dengan menggunakan Matlab dengan kode berikut

```
clc
clear all
close all
```

```
close all hidden
%
disp('Matrik A')
A = [5.0000  1.0000  3.0000; 1.0000  4.0000  6.0000; 3.0000
6.0000  8.0000]
disp('Dimensi Matrik A')
T = size(A)
```

Hasil program

```
Matrik A
A =
     5     1     3
     1     4     6
     3     6     8

Dimensi Matrik A
T =
     3     3
```

c. Matrik Diagonal

Matrik diagonal adalah matrik yang semua elemen bukan diagonal sama dengan nol.

Contoh 9.4: Suatu matrik diagonal dengan dimensi 3 x 3 dinyatakan dalam bentuk persamaan (9.16) berikut

$$A=\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 2.0000 & 0.0000 \\ 0.0000 & 0.0000 & 3.0000 \end{bmatrix} \quad \text{...} \quad (9.16)$$

Matrik pada persamaan (4.16) direpresentasikan dengan menggunakan Matlab dengan kode berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
T = [1.0000  2.0000   3.0000];
A = diag(T)
disp('Dimensi Matrik A')
T = size(A)
```

Hasil program

```
Matrik A
A =
     1     0     0
     0     2     0
     0     0     3

Dimensi Matrik A
T =
     3     3
```

d. Matrik Satuan

Matrik satuan adalah matrik diagonal yang semua elemen diagonal sama dengan 1.

Contoh 9.5: Suatu matrik diagonal dengan dimensi 3 x 3 dinyatakan dalam bentuk persamaan (9.17) berikut

$$A=\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \quad \text{.........} \quad (9.17)$$

Matrik pada persamaan (4.17) direpresentasikan dengan menggunakan Matlab dengan kode berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
T = [1.0000  1.0000   1.0000];
A = diag(T)
disp('Dimensi Matrik A')
T = size(A)
```

Hasil program

```
Matrik A
A =
     1     0     0
     0     1     0
     0     0     1
```

```
Dimensi Matrik A
T =
     3     3
```

e. Matrik Segitiga Atas

Matrik segitiga atas adalah matrik bujursangkar yang semua elemen di bawah diagonal sama dengan nol.

Contoh 9.6: Suatu matrik dengan dimensi 3 x 3 dinyatakan dalam bentuk persamaan (9.18) berikut

$$A=\begin{bmatrix} 10.0000 & 6.0000 & 3.0000 \\ 3.0000 & 4.0000 & 2.0000 \\ 4.0000 & 2.0000 & 5.0000 \end{bmatrix} \quad \ldots\ldots\ldots \quad (9.18)$$

Dengan menggunakan Matlab, tentukan matrik segitiga atas dari matrik yang dinyatakan pada persamaan (9.18). Adapun kode Matlab yang digunakan untuk memperoleh matrik segitiga atas sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [10.0000  6.0000  3.0000; 3.0000  4.0000  2.0000; 4.0000
2.0000  5.0000]
disp('Matrik Segitiga Atas')
Au = triu(A)
```

Hasil program

```
Matrik A
A =
    10     6     3
     3     4     2
     4     2     5

Matrik Segitiga Atas
Au =
    10     6     3
     0     4     2
     0     0     5
```

Matrik segitiga atas yang diperoleh diperlihatkan pada persamaan (9.19) berikut

$$A=\begin{bmatrix} 10.0000 & 6.0000 & 3.0000 \\ 0.0000 & 4.0000 & 2.0000 \\ 0.0000 & 0.0000 & 5.0000 \end{bmatrix} \quad \text{..........} \quad (9.19)$$

f. Matrik Segitiga Bawah

Matrik segitiga bawah adalah matrik bujursangkar yang semua elemen di atas diagonalnya sama dengan nol.

Contoh 9.7: Suatu matrik dengan dimensi 3 x 3 dinyatakan dalam bentuk persamaan (9.20) berikut

$$A=\begin{bmatrix} 10.0000 & 6.0000 & 3.0000 \\ 3.0000 & 4.0000 & 2.0000 \\ 4.0000 & 2.0000 & 5.0000 \end{bmatrix} \quad \text{..........} \quad (9.20)$$

Dengan menggunakan Matlab, tentukan matrik segitiga bawah dari matrik yang dinyatakan pada persamaan (9.20). Adapun kode Matlab yang digunakan untuk memperoleh matrik segitiga bawah sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [10.0000  6.0000  3.0000; 3.0000  4.0000  2.0000; 4.0000
2.0000  5.0000]
disp('Matrik Segitiga Bawah')
Au = tril(A)
```

Hasil program

```
Matrik A
A =

    10     6     3
     3     4     2
     4     2     5

Matrik Segitiga Bawah
Al =
```

```
    10      0      0
     3      4      0
     4      2      5
```

Matrik segitiga bawah yang diperoleh diperlihatkan pada persamaan (9.21) berikut

$$A=\begin{bmatrix} 10.0000 & 0.0000 & 0.0000 \\ 3.0000 & 4.0000 & 0.0000 \\ 4.0000 & 2.0000 & 5.0000 \end{bmatrix} \quad \dots\dots\dots \quad (9.21)$$

g. Matrik Pita

Matrik pita adalah matrik yang mempuyai elemen sama dengan nol, kecuali pada suatu pita yang berpusat pada diagonal.

Contoh 9.8: Suatu matrik dengan dimensi 5 x 5 dinyatakan dalam bentuk persamaan (9.22) berikut

$$A=\begin{bmatrix} 1.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0000 \\ 2.0000 & 4.0000 & 5.0000 & 0.0000 & 0.0000 \\ 0.0000 & 5.0000 & 1.0000 & 2.0000 & 0.0000 \\ 0.0000 & 0.0000 & 2.0000 & 3.0000 & 1.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 & 8.0000 \end{bmatrix} \quad \dots\dots\dots \quad (9.22)$$

Matrik pada persamaan (9.22) direpresentasikan dengan menggunakan Matlab dengan kode berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [1.0000  1.0000  0.0000  0.0000   0.0000;
     2.0000  4.0000  5.0000  0.0000   0.0000;
     0.0000  5.0000  1.0000  2.0000   0.0000;
     0.0000  0.0000  2.0000  3.0000   1.0000;
     0.0000  0.0000  0.0000  1.0000   8.0000]
```

Hasil program

```
Matrik A
A =
     1     1     0     0     0
     2     4     5     0     0
     0     5     1     2     0
     0     0     2     3     1

     0     0     0     1     8
```

Adapun lebar pita dari matrik A dari persamaan (9.22) adalah 3.

9.1.3. Operasi Baris Elementer

Metoda dasar untuk memecahkan sebuah sistem persamaan linier adalah dengan cara mengganti sistem yang diberikan dengan sistem yang baru, yang mempuyai himpunan pemecahan yang sama, tetapi lebih mudah untuk memecahkannya. Sistem baru ini umumnya didapatkan dari sederetan langkah operasi baris elementer yang digunakan untuk mengeliminasi bilangan – bilangan anu secara sistematis. Operasi baris elementer terdiri dari 3 langkah yaitu

a. Perkalian suatu baris dengan konstanta tak nol. Secara matematis dinyatakan dalam bentuk persamaan (9.23) berikut

$$b_i \rightarrow kb_i \quad \text{.......................} \quad (9.23)$$

b. Pergantian dua baris. Secara matematis dinyatakan dalam bentuk persamaan (9.24) berikut

$$b_i \rightarrow b_j \quad \text{.......................} \quad (9.24)$$

c. Suatu baris ditambahkan dengan kelipatan baris yang lain. Secara matematis dinyatakan dalam bentuk persamaan (9.25) berikut

$$b_i \rightarrow b_i + kb_j \quad \text{.......................} \quad (9.25)$$

Setiap operasi baris elementer tidak akan merubah nilai dari bilangan anu pada sistem persamaan linier tersebut.

Berbagai metoda dapat digunakan untuk memperoleh nilai x_1, x_2, ..., x_n diantaranya metoda eliminasi Gauss, metoda eliminasi Gauss – Jordan dan metoda Dekomposisi LU. Untuk metoda Dekomposisi LU ini terbagi atas metoda Crout, metoda Doolitle dan metoda Cholensky.

9.2 Eliminasi Gauss

Metoda ini merupakan suatu prosedure yang sistematis untuk memecahkan sistem persamaan linier yang didasarkan pada pemikiran untuk mereduksi matrik yang dilengkapi menjadi bentuk yang cukup sederhana. Prosedure ini menggunakan operasi baris elementer. Jika diketahui sistem persamaan linier dengan bentuk persamaan (9.1). Persamaan (9.1) kemudian diubah menjadi persamaan (9.7). Langkah selanjutnya adalah mengeliminasi persamaan (9.7) menjadi bentuk yang sederhana dengan operasi baris elementer. Hasil operasi baris elementer dari persamaan (9.7) diperlihatkan pada persamaan (9.26) berikut

$$AC = \begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} & c'_1 \\ 0 & a'_{22} & \cdots & a'_{2n} & c'_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & a'_{nn} & c'_n \end{bmatrix} \quad \text{..} \quad (9.26)$$

Langkah selanjutnya adalah menuliskan persamaan (9.26) menjadi persamaan (9.27) berikut

$$\begin{aligned} a'_{11}x'_1 + a'_{12}x'_2 + \ldots + a'_{1n}x'_n &= c'_1 \\ a'_{12}x'_2 + \ldots + a'_{2n}x'_n &= c'_2 \\ &\vdots \\ a'_n x'_n &= c'_n \end{aligned} \quad \text{..} \quad (9.27)$$

Akhirnya dengan substitusi mundur diperoleh persamaan (9.28) berikut

$$\begin{aligned} x_n &= \frac{c'_n}{a'_{nn}} \\ x_{n-1} &= \frac{c'_{n-1} - a'_{n-1,n}x'_n}{a'_{n-1,n-1}} \quad \text{..} \quad (9.28) \\ &\vdots \\ x_1 &= \frac{c'_1 - \sum_{2}^{n} a'_{ij}x'_j}{a_{11}} \end{aligned}$$

Dalam proses eliminasi Gauss sering ditemukan kondisi – kondisi berikut

a. Pembagian dengan nol
b. Galat pembulatan
c. Sistem berkondisi buruk

Bila keadaan tersebut terjadi akan menyebabkan penyelesaian tidak sesuai dengan yang dikehendaki. Salah satu cara untuk memperbaiki keadaan tersebut adalah proses Pivoting. Untuk memperbaiki penyelesaian sistem persamaan linier dengan proses Pivoting adalah dengan cara memilih elemen tumpuan yang nilai mutlaknya terbesar diantara koefesien yang tersedia. Misalkan pada langkah pertama, elemen yang nilai mutlaknya terbesar pada kolom pertama seperti yang dinyatakan dalam bentuk persamaan (9.29) berikut

$$\{|a_{1i}|\} = \text{maks}\{|a_{11}|, |a_{21}|, |a_{31}|, \ldots, |a_{n1}|\} \qquad (9.29)$$

Proses untuk Pivoting untuk langkah ke-1 diperlihatkan pada persamaan (9.30) berikut

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & c_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & c_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n1} & \cdots & a_{nn} & c_n \end{bmatrix} \qquad (9.30)$$

Pada metoda terdahulu, elemen tumpuan pada langkah ke -1 adalah a_{11}. Namun untuk metoda dengan proses pivoting dipilih elemen tumpuan dengan elemen terbesar pada kolom ke-1. Jika $|a_{11}|$ bukan nilai terbesar diantara elemen pada kolom ke-1, maka harus dilakukan penggantian baris sedemikian rupa sehingga $|a_{11}|$ merupakan nilai terbesar pada kolom ke-1. Contoh proses Pivoting langkah ke -1 ini diperlihatkan pada Contoh 9.9 berikut

Contoh 9.9: Lakukan proses Pivoting untuk sistem persamaan linier dalam bentuk persamaan (9.31) berikut

$$\begin{bmatrix} 1.0000 & 0.0000 & 3.0000 & 1.0000 \\ -1.0000 & 2.0000 & 1.0000 & 2.0000 \\ 4.0000 & -1.0000 & 3.0000 & 4.0000 \end{bmatrix} \qquad (9.31)$$

Pada persamaan (9.31) dilakukan proses Pivoting dengan memilih a_{31} = 4.0000 mempuyai nilai terbesar, sehingga untuk memperoleh elemen tumpuan a_{11} = 4.0000 baris ke -1 harus ditukar dengan baris ke-3.

Proses untuk Pivoting untuk langkah ke-2 diperlihatkan pada persamaan (9.32) berikut

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & c_1 \\ 0 & a_{22} & \cdots & a_{2n} & c_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & a_{n1} & \cdots & a_{nn} & c_n \end{bmatrix} \quad \text{..........} \quad (9.32)$$

Disini elemen tumpuan adalah a_{22}. Namun bila $|a_{22}|$ bukan nilai terbesar pada kolom ke-2 untuk baris 2 sampai ke-n, maka harus dilakukan penukaran baris sehingga akan diperoleh $|a_{22}|$ sebagai nilai terbesar. Contoh proses Pivoting langkah ke - 2 ini diperlihatkan pada Contoh 4.10 berikut

Contoh 9.10: Lakukan proses Pivoting untuk sistem persamaan linier dalam bentuk persamaan (9.33) berikut

$$\begin{bmatrix} 5.0000 & -1.0000 & 6.0000 & 9.0000 \\ 0.0000 & 2.0000 & 1.0000 & 0.0000 \\ 0.0000 & 3.0000 & 4.0000 & 5.0000 \end{bmatrix} \quad \text{..........} \quad (9.33)$$

Pada persamaan (9.33) dilakukan proses Pivoting dengan memilih $a_{22} = 3.0000$ sehingga untuk itu baris ke-3 dan baris ke-2 harus ditukar.

Contoh 9.11: Diketehui sistem persamaan linier dalam bentuk persamaan (9.34) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned} \quad \text{..........} \quad (9.34)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan eliminasi Gauss.

Jawab :

Persamaan (9.34) diselesaikan dengan menggunakan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
```

```
% Contoh 4.11
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000;
     2.0000  4.0000 -3.0000;
     3.0000  6.0000 -5.0000]
%
disp('Matrik b')
b = [9.0000; 1.0000; 0.0000]
%
n = size(A,1);
A = [A,b];
%
for i = 1:n-1
    p = i;
    for j = i+1:n
        if abs(A(j,i)) > abs(A(i,i))
            U = A(i,:);
            A(i,:) = A(j,:);
            A(j,:) = U;
        end
    end
    while A(p,i)== 0 & p <= n
        p = p+1;
    end
    if p == n+1
        disp('Tidak Ada Solusi Unik');
        break
    else
        if p ~= i
            T = A(i,:);
            A(i,:) = A(p,:);
            A(p,:) = T;
        end
    end

    for j = i+1:n
        m = A(j,i)/A(i,i);
        for k = i+1:n+1
            A(j,k) = A(j,k) - m*A(i,k);
        end
    end
end
%
if A(n,n) == 0
    disp('Tidak Ada Solusi Unik');
    return
```

```
end
%
x(n) = A(n,n+1)/A(n,n);
for i = n - 1:-1:1
    sumax = 0;
    for j = i+1:n
        sumax = sumax + A(i,j)*x(j);
    end
    x(i) = (A(i,n+1) - sumax)/A(i,i);
end
disp('Nilai x :')
x = x'
```

Hasil program

```
Matrik A
A =
     1     1     2
     2     4    -3
     3     6    -5
Matrik b
b =
     9
     1
     0
Nilai x :
x =
    1.0000
    2.0000
    3.0000
```

Adapun solusi persamaan (9.34) adalah x_1 = 1.0000, x_2 = 2.0000 dan x_3 = 3.0000.

9.3 Metoda Gauss Jordan

Metoda Gauss – Jordan adalah variasi dari metoda eliminasi Gauss. Dengan metoda eliminasi Gauss, koefesien matrik diubah menjadi bentuk matrik segitiga atas dan dengan metoda Gauss Jordan, koefesien matrik diubah menjadi bentuk matrik identitas dengan memodifikasi persamaan (9.3) menjadi persamaan (9.35) berikut

$$\begin{bmatrix} a_{11}^* & 0 & \cdots & 0 \\ 0 & a_{22}^* & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & a_{nn}^* \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} = \begin{bmatrix} c_1^* \\ c_2^* \\ \vdots \\ c_n^* \end{bmatrix} \quad \text{.........} \quad (9.35)$$

Bentuk lain dari persamaan (9.35) dinyatakan dalam bentuk persamaan (9.36) berikut

$$A_{nxn}^* x = c_n^* \quad \text{.........} \quad (9.36)$$

dimana A_{nxn}^* adalah matrik identitas yang dinyatakan dalam bentuk persamaan (9.37) berikut

$$a_{11}^* = a_{22}^* = \cdots = a_{nn}^* = 1 \quad \text{.........} \quad (9.37)$$

Solusi vektor x dapat diperoleh langsung yang dinyatakan dalam bentuk persamaan (9.38) berikut

$$\begin{aligned} x_1 &= c_1^* \\ x_2 &= c_2^* \\ &\vdots \\ x_n &= c_n^* \end{aligned} \quad \text{.........} \quad (9.38)$$

Selain itu dengan menggunakan notasi matriks, persamaan (9.35) dapat dituliskan dalam bentuk persamaan (9.39) berikut

$$Ix = c^* \quad \text{.........} \quad (9.39)$$

Secara komputasi metoda Gauss Jordan dapat diterapkan dengan 2 cara. Cara pertama dengan menggunakan matriks inversi dari A. Dengan cara ini matriks identitas I didapat dengan mengalikan matriks koefesien A dengan A^{-1}. Kolom vektor c^*diperoleh dengan cara mengalikan c dengan A^{-1}. Cara ini juga disebut dengan metoda inversi matriks. Cara kedua matrik A diubah menjadi matriks A^* dan vektor b menjadi vektor b^*. Perubahan ini dapat dilakukan sekaligus dalam proses komputasi dengan mengaugmentasikan matriks A dan vektor b. Secara komputasi matriks A dan b dapat diaugmentasikan menjadi matriks W dalam bentuk persamaan (9.40) berikut

$$W = [A \ \vdots \ b]_{n\,x\,(n+1)} \quad \text{.........} \quad (9.40)$$

Setelah itu dilakukan pengolahan matriks sehingga diperoleh matriks W^* yang mempuyai bentuk identitas pada bagian sebelah kiri garis putus – putus matriks W dalam bentuk persamaan (9.41) berikut

$$W^* = \left[A^* \;\; \vdots \;\; b^* \right]_{n\,x\,(n+1)} \quad \text{..........} \quad (9.41)$$

dimana A^* adalah matrik identitas dan b^*adalah vektor sisi kanan yang telah berubah sebagai konsekuensi dari perubahan matriks A. Proses perubahan matriks A menjadi A^*dengan metode ini sama dengan metoda eliminasi Gauss, hanya saja sekarang matriks A^* adalah matriks identitas karena A^* adalah matriks identitas maka b^* adalah solusi vektor x.

Contoh 9.12: Diketehui sistem persamaan linier dalam bentuk persamaan (9.42) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned} \quad \text{..........} \quad (9.42)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan eliminasi Gauss - Jordan.

Jawab :

Persamaan (9.42) diselesaikan dengan menggunakan Matlab dengan fungsi rref. Penerapan fungsi rref ini sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000   9.0000;
     2.0000  4.0000 -3.0000   1.0000;
     3.0000  6.0000 -5.0000   0.0000]
%
disp('Matrik R')
R = rref(A)
```

Hasil program

```
Matrik A
A =
     1     1     2     9
```

```
     2     4    -3     1
     3     6    -5     0
Matrik R
R =
     1     0     0     1
     0     1     0     2
     0     0     1     3
```

Adapun solusi persamaan (9.42) adalah x_1 = 1.0000, x_2 = 2.0000 dan x_3 = 3.0000.

Selain itu persamaan (9.42) dapat juga diselesaikan dengan kode Matlab sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000   9.0000;
     2.0000  4.0000 -3.0000   1.0000;
     3.0000  6.0000 -5.0000   0.0000]
%
tam = size(A);
fil=tam(1,1);
col=tam(1,2);
%
for k = 1:fil,
   may = abs(A(k,k));
   pos = k;
   y=A;
   for i=k+1:fil,
      if may < abs(A(i,k))
         may=abs(A(i,k));
         aux=A(i,:);
       A(i,:)= A(k,:);
       A(k,:)=aux;
      end
   end
   y=A;
   for i = 1:fil,
      if i ~= k
         pivote=A(i,k)/A(k,k);
         for j = k:col,
            A(i,j) = A(i,j) - pivote*A(k,j);
```

```
            end
         end
         y=A;
      end
   end
   %
   for i = 1:fil,
      A(i,col)= A(i,col)/A(i,i);
      A(i,i)=1;
   end
   y=A;
   %
   for i=1:fil,
      x(i)=A(i,col);
   end
   x
```

Hasil program

```
Matrik A
A =
     1     1     2     9
     2     4    -3     1
     3     6    -5     0
x =
    1.0000    2.0000    3.0000
```

Adapun solusi persamaan (9.42) adalah x_1 = 1.0000, x_2 = 2.0000 dan x_3 = 3.0000.

9.4 Metoda Gauss Seidel

Untuk sistem persamaan linier yang terdiri dari banyak persamaan dan tidak mempuyai bentuk tri-diagonal dan penta-diagonal, teknik – teknik langsung menjadi tidak efisien dalam komputasi, karena ukuran matrik yang besar. Untuk problem semacam ini teknik – teknik iterasi lebih efisien digunakan dalam komputasi. Teknik iterasi selalu dimulai dengan suatu vektor awal $x^{(0)}$ yang diasumsikan. Berdasarkan nilai vektor $x^{(0)}$ maka nilai vektor $x^{(1)}$. Nilai vektor $x^{(1)}$diharapkan lebih mendekati x yang merupakan vektor solusi. Dengan menggunakan nilai vektor $x^{(1)}$kemudian dihitung nilai vektor $x^{(2)}$. Nilai vektor $x^{(2)}$ diharapkan lebih baik daripada nilai vektor $x^{(1)}$. Iterasi ini akan dihentikan sampai nilai vektor $x^{(n)}$ yang diperoleh berada dalam toleransi yang diinginkan. Penentuan telah tercapai konvergen atau tidak merupakan keputusan yang dibuat. Perlu diketahui bahwa tidak semua sistem persamaan linier jika dipecahkan dengan teknik iterasi akan mendekati solusi sebenarnya atau konvergen. Syarat utama untuk sistem persamaan linier bisa konvergen adalah koefesien matriknya harus diagonal dominan. Suatu sistem yang tidak bersifat diagonal dominan dapat dijadikan bersifat diagonal dominan dengan mengatur posisi lajur. Selain itu jumlah iterasi untuk mencapai konvergen tergantung dari beberapa faktor diantaranya

1. Tingkat dominan dari koefesien pada diagonalnya
2. Vektor tebakan awal
3. Algoritma yang digunakan
4. Kriteria konvergensinya

Pada bagian ini akan dibahas algoritma iterasi dengan metoda Iterasi Gauss Seidel. Proses iterasi metoda Gauss Seidel yang dilakukan dinyatakan dalam bentuk persamaan (9.43) berikut

$$
\begin{aligned}
x_1^{(k+1)} &= -\frac{1}{a_{11}}\left(a_{12}x_2^{(k)} + a_{13}x_3^{(k)} + a_{14}x_4^{(k)} + \ldots + a_{1n}x_n^{(k)} - b_1\right) \\
x_2^{(k+1)} &= -\frac{1}{a_{22}}\left(a_{21}x_1^{(k)} + a_{23}x_3^{(k)} + a_{24}x_4^{(k)} + \ldots + a_{2n}x_n^{(k)} - b_2\right) \\
&\vdots \\
x_n^{(k+1)} &= -\frac{1}{a_{nn}}\left(a_{n1}x_1^{(k)} + a_{n2}x_3^{(k)} + a_{n3}x_4^{(k)} + \ldots + a_{n(n-1)}x_n^{(k+1)} - b_n\right)
\end{aligned}
\qquad \text{..........} \qquad (9.43)
$$

Secara umum persamaan (9.43) dapat dituliskan dalam bentuk persamaan (9.44) berikut

$$
x_i^{(k+1)} = -\frac{1}{a_{ii}}\left(\sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} + \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} - b_i\right) \qquad \text{..........} \qquad (9.44)
$$

Atau dapat juga dituliskan dengan menambah kemudian mengurangi sisi kanan persamaan (9.44) dengan $x_i^{(k)}$ sehingga diperoleh persamaan (9.45) berikut

$$
x_i^{(k+1)} = x_i^{(k)} + \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i}^{n} a_{ij}x_j^{(k)}\right) \qquad \text{..........} \qquad (9.45)
$$

Proses iterasi dihentikan apabila memenuhi persamaan (9.46) berikut

$$
\begin{aligned}
&\left|x_i^{(k+1)} - x_i^{(k)}\right| \le \\
&\left|\frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i}^{n} a_{ij}x_j^{(k)}\right)\right| \le
\end{aligned}
\qquad \text{..........} \qquad (9.46)
$$

Selain itu berdasarkan persamaan (9.46) diperoleh persamaan (9.47) berikut

$$r_i^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i}^{n} a_{ij} x_j^{(k)} \quad \text{..} \quad (9.47)$$

Dimana å adalah nilai toleransi konvergensi dan $r_i^{(k)}$ adalah residu perhitungan. Berdasarkan persamaan (9.47) maka persamaan (9.45) berubah menjadi persamaan (9.48) berikut

$$x_i^{(k+1)} = x_i^{(k)} + \frac{r_i^{(k)}}{a_{ii}} \quad \text{..} \quad (9.48)$$

Dengan demikian kriteria untuk menghentikan proses iterasi dalam metoda Gauss Seidel adalah

1. Apabila jumlah iterasi telah mencapai maksimum iterasi yang telah ditentukan.

2. Apabila perbedaan antara $\left| \frac{r_i^{(k)}}{a_{ii}} \right| \le \mathring{a}$ untuk semua nilai x_i

Contoh 9.13: Diketehui sistem persamaan linier dalam bentuk persamaan (9.49) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned} \quad \text{...} \quad (9.49)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan eliminasi Gauss – Seidel.

Jawab :

Persamaan (9.49) diselesaikan dengan bantuan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
format compact
%
disp('Matrik A')
A = [1  1  2;2  4  -3;3  6  -5]
```

```
%
disp('Matrik b')
C = [9 ; 1 ; 0]
%
n = length(C);
X = zeros(n,1);
Error_eval = ones(n,1);
%
for i = 1:n
    j = 1:n;
    j(i) = [];
    B = abs(A(i,j));
    Check(i) = abs(A(i,i)) - sum(B);
    if Check(i) < 0
    end
end
iteration = 0;
while max(Error_eval) > 0.001
    iteration = iteration + 1;
    Z = X;
    for i = 1:n
        j = 1:n;
        j(i) = [];
        Xtemp = X;
        Xtemp(i) = [];
        X(i) = (C(i) - sum(A(i,j) * Xtemp)) / A(i,i);
    end
    Xsolution(:,iteration) = X;
    Error_eval = sqrt((X - Z).^2);
end
disp(' ')
disp('Proses Iterasi')
K = [1:iteration;Xsolution]'
%
disp(' ')
disp(['Jumlah Iterasi      : ',num2str(iteration)])
disp(' ')
disp(['Tingkat Presisi     : '])
Error_eval
disp(' ')
disp('Nilai X')
X
```

Hasil program

```
Matrik A
A =
     1     1     2
     2     4    -3
     3     6    -5
Matrik b
C =
     9
     1
     0

Proses Iterasi
K =
    1.0000    9.0000   -4.2500    0.3000
    2.0000   12.6500   -5.8500    0.5700
    3.0000   13.7100   -6.1775    0.8130
    4.0000   13.5515   -5.9160    1.0317
    5.0000   12.8526   -5.4025    1.2285
    6.0000   11.9455   -4.8013    1.4057
    7.0000   10.9900   -4.1907    1.5651
    8.0000   10.0605   -3.6064    1.7086
    9.0000    9.1892   -3.0632    1.8377
   10.0000    8.3877   -2.5655    1.9540
   11.0000    7.6576   -2.1133    2.0586
   12.0000    6.9962   -1.7042    2.1527
   13.0000    6.3987   -1.3348    2.2374
   14.0000    5.8600   -1.0019    2.3137
   15.0000    5.3745   -0.7020    2.3823
   16.0000    4.9373   -0.4319    2.4441
   17.0000    4.5437   -0.1888    2.4997
   18.0000    4.1894    0.0301    2.5497
   19.0000    3.8705    0.2270    2.5947
   20.0000    3.5835    0.4043    2.6353
   21.0000    3.3251    0.5639    2.6717
   22.0000    3.0926    0.7075    2.7046
   23.0000    2.8834    0.8367    2.7341
   24.0000    2.6950    0.9531    2.7607
   25.0000    2.5255    1.0578    2.7846
   26.0000    2.3730    1.1520    2.8062
   27.0000    2.2357    1.2368    2.8256
   28.0000    2.1121    1.3131    2.8430
   29.0000    2.0009    1.3818    2.8587
   30.0000    1.9008    1.4436    2.8728
   31.0000    1.8107    1.4993    2.8855
   32.0000    1.7297    1.5493    2.8970
   33.0000    1.6567    1.5944    2.9073
```

```
   34.0000    1.5910    1.6350    2.9166
   35.0000    1.5319    1.6715    2.9249
   36.0000    1.4787    1.7043    2.9324
   37.0000    1.4309    1.7339    2.9392
   38.0000    1.3878    1.7605    2.9453
   39.0000    1.3490    1.7844    2.9507
   40.0000    1.3141    1.8060    2.9557
   41.0000    1.2827    1.8254    2.9601
   42.0000    1.2544    1.8429    2.9641
   43.0000    1.2290    1.8586    2.9677
   44.0000    1.2061    1.8727    2.9709
   45.0000    1.1855    1.8854    2.9738
   46.0000    1.1669    1.8969    2.9764
   47.0000    1.1502    1.9072    2.9788
   48.0000    1.1352    1.9165    2.9809
   49.0000    1.1217    1.9248    2.9828
   50.0000    1.1095    1.9324    2.9845
   51.0000    1.0986    1.9391    2.9861
   52.0000    1.0887    1.9452    2.9875
   53.0000    1.0798    1.9507    2.9887
   54.0000    1.0719    1.9556    2.9899
   55.0000    1.0647    1.9601    2.9909
   56.0000    1.0582    1.9641    2.9918
   57.0000    1.0524    1.9676    2.9926
   58.0000    1.0471    1.9709    2.9933
   59.0000    1.0424    1.9738    2.9940
   60.0000    1.0382    1.9764    2.9946
   61.0000    1.0344    1.9788    2.9951
   62.0000    1.0309    1.9809    2.9956
   63.0000    1.0278    1.9828    2.9961
   64.0000    1.0251    1.9845    2.9965
   65.0000    1.0225    1.9861    2.9968
   66.0000    1.0203    1.9875    2.9971
   67.0000    1.0183    1.9887    2.9974
   68.0000    1.0164    1.9898    2.9977
   69.0000    1.0148    1.9909    2.9979
   70.0000    1.0133    1.9918    2.9981
   71.0000    1.0120    1.9926    2.9983
   72.0000    1.0108    1.9933    2.9985
   73.0000    1.0097    1.9940    2.9986
   74.0000    1.0087    1.9946    2.9988

Jumlah Iterasi      : 74

Tingkat Presisi     :
Error_eval =
   1.0e-03 *
```

```
    0.9706
    0.5995
    0.1370

Nilai X
X =
    1.0087
    1.9946
    2.9988
```

Adapun solusi persamaan (9.49) adalah x_1 = 1.0087, x_2 = 1.9946 dan x_3 = 2.9988.

9.5 Metoda Dekomposisi LU

Setiap persoalan yang melibatkan sistem persamaan linear dapat dituliskan dengan notasi matrik dalam bentuk persamaan (9.50) berikut

$$Ax = b \quad \text{.......} \quad (9.50)$$

Matrik A pada persamaan (9.50) merupakan perkalian dari matrik segitiga bawah (L) dan matrik segitiga atas (U) yang dinyatakan dalam bentuk persamaan (9.51) berikut

$$A = LU \quad \text{.......} \quad (9.51)$$

Matrik segitiga bawah (L) dan matrik segitiga atas (U) ini diperoleh dengan menggunakan metoda Doolitle, metoda Crout dan metoda Cholensky. Selain itu persamaan (9.51) ini disubstitusikan ke persamaan (9.50) dan diperoleh persamaan (9.52) berikut

$$LUx = b \quad \text{.......} \quad (9.52)$$

Berdasarkan persamaan (9.52) diperoleh persamaan (9.53) dan (9.54) berikut

$$Ly = b \quad \text{.......} \quad (9.53)$$

$$Ux = y \quad \text{.......} \quad (9.54)$$

Contoh 9.14: Diketehui sistem persamaan linier dalam bentuk persamaan (9.48) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \quad \text{.......} \quad (9.55) \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned}$$

Dengan menggunakan Matlab, tentukan matrik segitiga bawah (L) dan matrik segitiga atas (U) dari persamaan (9.55)

Jawab :

Persamaan (9.55) diselesaikan bantuan Matlab dengan menggunakan fungsi lu. Penerapan fungsi lu dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000;
     2.0000  4.0000 -3.0000;
     3.0000  6.0000 -5.0000]
%
disp('Dekomposisi LU')
[L,U] = lu(A)
```

Hasil program

```
Matrik A
A =
     1     1     2
     2     4    -3
     3     6    -5

Dekomposisi LU
L =
    0.3333    1.0000         0
    0.6667         0    1.0000
    1.0000         0         0
U =
    3.0000    6.0000   -5.0000
         0   -1.0000    3.6667
         0         0    0.3333
```

9.5.1 Metoda Doolitle

Metoda Doolitle ini dapat digunakan untuk menyelesaikan sistem persamaan linier dengan beberapa vektor kolom b. Pada metoda Doolitle ini koefesien matrik A didekomposisikan menjadi dua matrik yang mempuyai bentuk matrik segitiga bawah (L) dan matrik segitiga atas (U). Pada bagian ini matrik segitiga bawah (L) dan matrik segitiga atas (U) dihitung dengan menggunakan metoda Doolitle dalam bentuk persamaan (9.56) berikut

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{24} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & a_{34} & \cdots & a_{3n} \\ a_{41} & a_{42} & a_{43} & a_{44} & \cdots & a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & 0 & & 0 \\ l_{31} & l_{32} & 1 & & & 0 \\ l_{41} & l_{42} & l_{43} & 1 & & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ l_{n1} & l_{n1} & l_{n3} & l_{n4} & \cdots & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & u_{24} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & u_{34} & \cdots & u_{3n} \\ 0 & 0 & 0 & u_{44} & \cdots & u_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & u_{nn} \end{bmatrix} \quad (9.56)$$

Secara umum prosedur perhitungan dengan metoda Doolitle ini dinyatakan dalam bentuk persamaan (9.57) s/d (9.60) berikut

$$u_{ij} = a_{ij} \quad \text{untuk} \quad i = 1 \text{ dan } j = i, n \quad \ldots\ldots \quad (9.57)$$

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \quad \text{untuk} \quad i = 2, n \quad \text{dan } j = i, n \quad \ldots\ldots \quad (9.58)$$

$$l_{ij} = \frac{a_{ij}}{u_{jj}} \quad \text{untuk} \quad i = 1, n \quad \text{dan} \quad j = 1 \quad \ldots\ldots \quad (9.59)$$

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj}}{u_{jj}} \quad \text{untuk} \quad i = 1, n \quad \text{dan } j = 2, i - 1 \quad \ldots\ldots \quad (9.60)$$

Contoh 9.15: Diketehui sistem persamaan linier dalam bentuk persamaan (9.61) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned} \quad \ldots\ldots \quad (9.61)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan metoda dekomposisi algoritma Doolitle.

Jawab :

Persamaan (9.61) diselesaikan dengan bantuan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
format compact
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000;
     2.0000  4.0000 -3.0000;
     3.0000  6.0000 -5.0000]
%
disp('Matrik b')
B = [9.0000 ; 1.0000 ; 0.0000]
%
[n,n]=size(A);
U = A;
L = zeros(n,n);
for i = 1 : n-1;
    if abs(U(i,i))> 0
        for i1 = i+1:n;
            m(i1,i) = U(i1,i)/U(i,i);
            for j = 1 : n
                U(i1,j) = U(i1,j) - m(i1,i)*U(i,j);
            end;
        end;
    end;
end;
for i = 1 :n;
    L(i,1) = A(i,1)/U(1,1);
end
for j = 2:n;
    for i = 2:n;
        s = 0;
        for k = 1:j-1;
            s = s + L(i,k)*U(k,j);
        end;
        L(i,j) = (A(i,j) - s)/U(j,j);
    end;
end;
y(1) = B(1)/L(1,1);
for k = 2:n;
    sum = B(k);
    for i = 1 : k - 1;
```

```
        sum = sum - L(k,i)*y(i);
    end;
    y(k) = sum/L(k,k);
end;
x(n) = y(n)/U(n,n);
for k = n-1:-1:1;
    sum = y(k);
    for i = k+1:n;
        sum = sum - U(k,i)*x(i);
    end;
    x(k) = sum/U(k,k);
end;
disp('Matrik Segitiga Bawah')
L
disp('Matrik Segitiga Atas')
U
disp('Nilai X')
X = x'
```

Hasil program

```
Matrik A
A =
     1     1     2
     2     4    -3
     3     6    -5
Matrik b
B =
     9
     1
     0
Matrik Segitiga Bawah
L =
    1.0000         0         0
    2.0000    1.0000         0
    3.0000    1.5000    1.0000
Matrik Segitiga Atas
U =
    1.0000    1.0000    2.0000
         0    2.0000   -7.0000
         0         0   -0.5000
Nilai X
X =
     1
     2
     3
```

Adapun solusi persamaan (9.61) adalah x_1 = 1.0000, x_2 = 2.0000 dan x_3 = 3.0000.

9.5.2 Metoda Crout

Metoda Crout ini dapat digunakan untuk menyelesaikan sistem persamaan linier dengan beberapa vektor kolom b. Pada metoda Crout ini koefesien matrik A didekomposisikan menjadi dua matrik yang mempuyai bentuk matrik segitiga bawah (L) dan matrik segitiga atas (U). Pada bagian ini matrik segitiga bawah (L) dan matrik segitiga atas (U) dihitung dengan menggunakan metoda Crout dalam bentuk persamaan (9.62) berikut

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{24} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & a_{34} & \cdots & a_{3n} \\ a_{41} & a_{42} & a_{43} & a_{44} & \cdots & a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & 0 & & 0 \\ l_{31} & l_{32} & l_{33} & & & 0 \\ l_{41} & l_{42} & l_{43} & l_{44} & & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ l_{n1} & l_{n1} & l_{n3} & l_{n4} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & u_{13} & u_{14} & \cdots & u_{1n} \\ 0 & 1 & u_{23} & u_{24} & \cdots & u_{2n} \\ 0 & 0 & 1 & u_{34} & \cdots & u_{3n} \\ 0 & 0 & 0 & 1 & \cdots & u_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{9.62}$$

Secara umum prosedur perhitungan dengan metoda Crout ini dinyatakan dalam bentuk persamaan (9.63) s/d (9.66) berikut

$$l_{i1} = a_{i1} \quad \text{untuk} \quad i = 1, n \text{ dan } j = 1 \quad \ldots\ldots \tag{9.63}$$

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \quad \text{untuk} \quad i = 2, n \text{ dan } j = 1, i \quad \ldots\ldots \tag{9.64}$$

$$u_{1j} = \frac{a_{1j}}{l_{11}} \quad \text{untuk} \quad i = 1 \quad \text{dan } j = 1, n \quad \ldots\ldots \tag{9.65}$$

$$u_{ij} = \frac{a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}}{l_{ii}} \quad \text{untuk} \quad i = 1, j - 1 \text{ dan } j = 3, n \quad \ldots\ldots \tag{9.66}$$

Contoh 9.16: Diketehui sistem persamaan linier dalam bentuk persamaan (9.67) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 2.0000x_3 &= 9.0000 \\ 2.0000x_1 + 4.0000x_2 - 3.0000x_3 &= 1.0000 \\ 3.0000x_1 + 6.0000x_2 - 5.0000x_3 &= 0.0000 \end{aligned} \quad \ldots\ldots \tag{9.67}$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan metoda dekomposisi algoritma Crout.

Jawab :

Persamaan (9.67) diselesaikan dengan bantuan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
format compact
%
disp('Matrik A')
A = [1.0000  1.0000  2.0000;
     2.0000  4.0000 -3.0000;
     3.0000  6.0000 -5.0000]
%
disp('Matrik b')
B = [9.0000 ; 1.0000 ; 0.0000]
%
n=rank(A);
%
L=zeros(n);
U=eye(n);
for s=1:n
j=s;
for i=j:n
    L(i,j)=A(i,j)-L(i,1:(j-1))*U(1:(j-1),j);
end
i=s;
U(i,i)=1;
for j=i+1:n
    U(i,j)=(A(i,j)-L(i,1:(i-1))*U(1:(i-1),j))/(L(i,i));
end
end
%
disp(' ')
disp('Periksa Matrik A')
check=int8 (L*U)
if check==A
    disp(sprintf ('Matrik L*U=A Benar  '))
else
    disp(sprintf ('Ada Kesalahan'))
end
```

```
%
xstar_soln(n)=0;
xstar_soln=xstar_soln';
xstar_soln(1)=B(1)/L(1,1);
for i=2:n
    xstar_soln(i)=(B(i,1)-L(i,1:i-1)*xstar_soln(1:i-1))/L(i,i);
end
%
x_soln(n)=0;
x_soln=x_soln';
x_soln(n)=xstar_soln(n);
i=n-1;
%
while i>0
    x_soln(i)=xstar_soln(i)-U(i,i+1:n)*x_soln(i+1:n);
   i=i-1;
end
disp(' ')
disp('Matrik Segitiga Bawah')
L
disp('Matrik Segitiga Atas')
U
disp('Nilai X')
X = x_soln
```

Hasil program

```
Matrik A
A =
     1     1     2
     2     4    -3
     3     6    -5
Matrik b
B =
     9
     1
     0

Periksa Matrik A
check =
    1    1    2
    2    4   -3
    3    6   -5
Matrik L*U=A Benar

Matrik Segitiga Bawah
L =
```

```
    1.0000         0         0
    2.0000    2.0000         0
    3.0000    3.0000   -0.5000
Matrik Segitiga Atas
U =
    1.0000    1.0000    2.0000
         0    1.0000   -3.5000
         0         0    1.0000
Nilai X
X =
     1
     2
     3
```

Adapun solusi persamaan (9.61) adalah x_1 = 1.0000, x_2 = 2.0000 dan x_3 = 3.0000.

9.5.3 Metoda Cholensky

Untuk sistem persamaan linier yang dinyatakan dengan persamaan (9.50) dimana koefesien matrik A yang simetris, metoda Cholensky dapat digunakan untuk memecahkan sistem persamaan linier dengan lebih efisien jika dibandingkan dengan metoda yang lain. Dengan metoda Cholensky ini, matrik A didekomposisikan menjadi matrik L dan matrik L^T. Metoda ini lebih menghemat memori karena hanya satu matrik yang perlu disimpan yaitu matrik L. Dengan menggunakan metoda Cholesky, matrik A dinyatakan dalam bentuk persamaan (9.68) berikut

$$A = LL^T \quad \text{..........} \qquad (9.68)$$

Selain itu persamaan (9.68) dijabarkan dalam bentuk persamaan (9.69) berikut

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{24} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & a_{34} & \cdots & a_{3n} \\ a_{41} & a_{42} & a_{43} & a_{44} & \cdots & a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & 0 & & 0 \\ l_{31} & l_{32} & l_{33} & & & 0 \\ l_{41} & l_{42} & l_{43} & l_{44} & & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ l_{n1} & l_{n1} & l_{n3} & l_{n4} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} & l_{41} & \cdots & l_{n1} \\ 0 & l_{22} & l_{32} & l_{42} & \cdots & l_{n2} \\ 0 & 0 & l_{33} & l_{32} & \cdots & l_{n3} \\ 0 & 0 & 0 & l_{44} & \cdots & l_{n4} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & l_{nn} \end{bmatrix} \qquad (9.69)$$

Secara umum prosedur perhitungan dengan metoda Cholensky ini dinyatakan dalam bentuk persamaan (9.70) s/d (9.73) berikut

$$l_{ii} = \sqrt{a_{ii}} \quad \text{untuk} \quad i = 1 \quad \ldots\ldots \quad (9.70)$$

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2} \quad \text{untuk} \quad i = 2, 3, \ldots, n \quad \ldots\ldots \quad (9.71)$$

$$l_{ij} = \frac{a_{ij}}{l_{jj}} \quad \text{untuk} \quad i = 2, 3, \ldots, n \quad \text{dan } j = 1 \quad \ldots\ldots \quad (9.72)$$

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{kj}}{l_{jj}} \quad \text{untuk} \quad i = 2, 3, \ldots, n \text{ , } j = 2, 3, \ldots, i\text{-}1 \text{ dan } k = 1, 2, \ldots, i\text{-}1 \quad (9.73)$$

Contoh 9.17: Diketehui sistem persamaan linier dalam bentuk persamaan (9.74) berikut

$$\begin{aligned}
6.0000x_1 + 15.0000x_2 + 55.0000x_3 &= 76.0000 \\
15.0000x_1 + 55.0000x_2 + 225.0000x_3 &= 295.0000 \\
55.0000x_1 + 225.0000x_2 + 9789.0000x_3 &= 1259.0000
\end{aligned} \quad \ldots\ldots \quad (9.74)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan metoda Cholensky.

Jawab :

Persamaan (9.74) diselesaikan dengan bantuan Matlab dengan kode sebagai berikut

```
clc
clear all
close all
close all hidden
%
A = [ 6  15  55;15  55   225;55 225 979]
B = [ sum(A(1,:)); sum(A(2,:)); sum(A(3,:))]
%
U = chol(A);
C = U'*U
D = U'\B
X = U\D
```

Hasil program

```
A =
     6    15    55
    15    55   225
    55   225   979

B =
          76
         295
        1259
X =
    1.0000
    1.0000
    1.0000
```

Adapun solusi persamaan (9.74) adalah $x_1 = 1.0000$, $x_2 = 1.0000$ dan $x_3 = 1.0000$.

9.6 Rangkuman

Sistem persamaan linier sering muncul dalam banyak permasalahan dibidang keteknikan, sains, operasi manajemen, social ekonomi dan bidang – bidang yang lain. Oleh karenanya pemecahan sistem persamaan linier merupakan salah satu teknik yang mutlak harus bisa diterapkan untuk memecahkan persoalan matematik secara numerik. Beberapa metoda yang dibahas meliputi eliminasi Gauss, eliminasi Gauss Jordan, metoda Gauss Seidel, metoda dekomposisi LU dengan algoritma Doolitle, metoda dekomposisi LU dengan algoritma Crout dan metoda dekomposisi dengan algoritma Cholensky.

Metoda eliminasi Gauss merupakan metoda yang melibatkan dua tahap komputasi yaitu proses pembentukan matrik segitiga atas dan proses substitusi mundur. Metoda eliminasi Gauss Jordan adalah variasi dari metoda eliminasi Gauss dimana metoda eliminasi Gauss mengubah koefesien matrik menjadi suatu matrik berbentuk segitiga atas maka metoda eliminasi Gauss Jordan mengubah matrik koefesien menjadi matrik identitas. Metoda Gauss Seidel merupakan salah satu metoda teknik iterasi yang sering digunakan dalam pemecahan sistem persamaan linier yang terdiri banyak persamaan dan tidak mempuyai bentuk tri –diagonal atau penta – diagonal dimana akan mengakibatkan tidak efisien dalam proses komputasinya. Untuk metoda dekomposisi merupakan metoda yang bisa digunakan untuk penyelesaian sistem persamaan linier dengan beberapa vektor kolom. Pada metoda ini, koefesien matrik A didekomposisi menjadi menjadi dua matrik yang mempuyai bentuk matrik segitiga bawah dan matrik segitiga atas. Penentuan matrik segitiga bawah dan matrik segitiga atas ini dilakukan dengan menggunakan algoritma Doolite, algoritma Crout dan algoritma Cholensky.

9.7 Soal - Soal

Soal 9.1: Diketehui sistem persamaan linier dalam bentuk persamaan (9.75) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 3.0000x_3 &= 2.0000 \\ 5.0000x_1 + 3.0000x_2 + 1.0000x_3 &= 3.0000 \\ 2.0000x_1 + 3.0000x_2 + 1.0000x_3 &= -1.0000 \end{aligned} \quad \text{..} \quad (9.75)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan eliminasi Gauss.

Soal 9.2: Diketehui sistem persamaan linier dalam bentuk persamaan (9.76) berikut

$$\begin{aligned} -2.0000x_1 + 3.0000x_2 + 1.0000x_3 &= 9.0000 \\ 3.0000x_1 + 4.0000x_2 - 5.0000x_3 &= 1.0000 \\ 1.0000x_1 - 2.0000x_2 + 1.0000x_3 &= -4.0000 \end{aligned} \quad \text{..} \quad (9.76)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan eliminasi Gauss - Jordan.

Soal 9.3: Diketehui sistem persamaan linier dalam bentuk persamaan (9.76) berikut

$$\begin{aligned} 2.0000x_1 - 1.0000x_2 + 1.0000x_3 &= -1.0000 \\ 3.0000x_1 + 3.0000x_2 + 9.0000x_3 &= 0.0000 \\ 3.0000x_1 + 3.0000x_2 + 5.0000x_3 &= 4.0000 \end{aligned} \quad \text{..} \quad (9.77)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan metoda iterasi Gauss Seidel dengan $x^{(0)} = 0.0000$.

Soal 9.4: Diketehui sistem persamaan linier dalam bentuk persamaan (9.78) berikut

$$\begin{aligned} 1.0000x_1 + 1.0000x_2 + 3.0000x_3 &= 4.0000 \\ 5.0000x_1 + 3.0000x_2 + 1.0000x_3 &= 6.0000 \\ 2.0000x_1 + 3.0000x_2 + 1.0000x_3 &= -1.0000 \end{aligned} \quad \text{...} \quad (9.78)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan dekomposisi LU dengan algoritma Doolitle.

Soal 9.5: Diketehui sistem persamaan linier dalam bentuk persamaan (9.79) berikut

$$\begin{aligned} 2.0000x_1 + 3.0000x_2 + 1.0000x_3 &= 7.0000 \\ 1.0000x_1 + 4.0000x_2 + 2.0000x_3 &= 2.0000 \\ 1.0000x_1 - 2.0000x_2 + 1.0000x_3 &= 4.0000 \end{aligned} \quad \text{...} \quad (9.79)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan dekomposisi LU dengan algoritma Crout.

Soal 9.6: Diketehui sistem persamaan linier dalam bentuk persamaan (9.76) berikut

$$\begin{aligned} 2.0000x_1 + 3.0000x_2 + 1.0000x_3 &= 7.0000 \\ 1.0000x_1 + 4.0000x_2 + 2.0000x_3 &= 2.0000 \\ 1.0000x_1 - 2.0000x_2 + 1.0000x_3 &= 4.0000 \end{aligned} \quad \text{...} \quad (9.80)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan dekomposisi LU dengan algoritma Crout.

Soal 9.7: Diketehui sistem persamaan linier dalam bentuk persamaan (9.81) berikut

$$\begin{aligned} 6.0000x_1 + 15.0000x_2 + 55.0000x_3 &= 76.0000 \\ 15.0000x_1 + 55.0000x_2 + 225.0000x_3 &= 295.0000 \\ 55.0000x_1 + 225.0000x_2 + 9789.0000x_3 &= 1259.0000 \end{aligned} \quad \text{.........................} \quad (9.81)$$

Dengan menggunakan Matlab, tentukan solusi sistem persamaan linier dengan menggunakan metoda Cholensky.

BAB X
INTEGRAL NUMERIK

10.1 Pendahuluan

Pada bagian ini dibahas tentang integral numerik. Integral numerik adalah suatu cara untuk menghitung aproksimasi luas daerah di bawah fungsi yang dimaksud pada selang yang diberikan. Beberapa metoda integrasi numerik yang sering digunakan adalah integral numerik dengan aturan Riemann, integral numerik dengan aturan Trapezoid dan integral numerik dengan aturan Simpson.

10.2 Integral Numerik Dengan Aturan Riemann

Integral numerik dengan aturan Riemann dinyatakan dalam bentuk persamaan (10.1) atau (10.2) berikut

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n-1} h\,f(x_i) \quad \text{......} \quad (10.1)$$

$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{n} h\,f(x_i) \quad \text{......} \quad (10.2)$$

dimana

$$h = \frac{b-a}{n} \quad \text{......} \quad (10.3)$$

Selain itu persamaan (10.1) dikenal juga sebagai penyelesaian integral numerik dengan aturan Riemann kiri dan persamaan (10.2) dikenal sebagai penyelesaian integral numerik dengan aturan Riemann kanan. Selain itu persamaan (10.1) dan (10.2) dapat juga dinyatakan dalam bentuk persamaan (10.4) berikut

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = h\,f(x_i) + O(h^2) \quad \text{......} \quad (10.4)$$

Contoh 10.;1: Dengan menggunakan Maltab dan aturan Riemann kiri, tentukan nilai integral dari persamaan (10.5) berikut

$$\int_0^{ð} \sin(x)\,dx \quad \text{......} \quad (10.5)$$

Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (10.5) dengan aturan Riemann kiri sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Integral Numerik Dengan Metoda Riemann Kiri
a = 0;
b = pi;
n = 11;
h = (b-a)/(n-1);
x = linspace(a,b,n);
f = sin(x);
%
I_riemann_L   = h*sum(f(1:n-1))
err_riemann_L = 2 - I_riemann_L
```

Hasil program

```
I_riemann_L =
    1.9835

err_riemann_L =
    0.0165
```

Hasil perhitungan dengan menggunskan Matlab diperoleh hasil perhitungan integral numerik persamaan (10.5) sebesar 1.9835 dengan tingkat kesalahan sebesar 0.0165

Contoh 10.2: Dengan menggunakan Maltab dan aturan Riemann kanan, tentukan nilai integral dari persamaan (10.6) berikut

$$\int_0^{\eth} \sin(x)\, dx \quad \text{..........} \quad (10.6)$$

Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (10.6) dengan aturan Riemann kanan sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Integral Numerik Dengan Metoda Riemann Kanan
a = 0;
```

```
b = pi;
n = 11;
h = (b-a)/(n-1);
x = linspace(a,b,n);
f = sin(x);
%
I_riemann_R   = h*sum(f(2:n))
err_riemann_R = 2 - I_riemann_R
```

Hasil program

```
I_riemann_R =
    1.9835

err_riemann_R =
    0.0165
```

Hasil perhitungan dengan menggunskan Matlab diperoleh hasil perhitungan integral numerik persamaan (10.6) sebesar 1.9835 dengan tingkat kesalahan sebesar 0.0165

10.3 Integral Numerik Dengan Aturan Titik Tengah

Integral numerik dengan aturan titik tengah dinyatakan dalam bentuk persamaan (10.7) dan (10.8) berikut

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n-1} h\,f(y_i) \qquad (10.7)$$

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = h\,f(y_i) + O(h^3) \qquad (10.8)$$

dimana

$$y_i = \frac{x_{i+1} + x_i}{2} \qquad (10.9)$$

Contoh 10.3: Dengan menggunakan Maltab dan aturan titik tengah, tentukan nilai integral dari persamaan (10.10) berikut

$$\int_0^{\eth} \sin(x)\,dx \qquad (10.10)$$

Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (10.10) dengan aturan titik tengah sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Integral Numerik Dengan Aturan Titik Tengah
a = 0;
b = pi;
n = 11;
h = (b-a)/(n-1);
x = linspace(a,b,n);
f = sin(x);
%
I_mid   = h*sum(sin((x(1:n-1) + x(2:n))/2))
err_mid = 2 - I_mid
```

Hasil program

```
I_mid =
    2.0082

err_mid =
   -0.0082
```

Hasil perhitungan dengan menggunakan Matlab diperoleh hasil perhitungan integral numerik persamaan (10.10) sebesar 2.0082 dengan tingkat kesalahan sebesar 0.0082

10.4 Integral Numerik Dengan Aturan Trapezoid

Integral numerik dengan aturan Trapezoid dinyatakan dalam bentuk persamaan (10.11) dan (10.12) berikut

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n-1} h\,\frac{f(x_i)+f(x_{i+1})}{2} \qquad (10.11)$$

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = h\,\frac{f(x_i)+f(x_{i+1})}{2} + O(h^3) \qquad (10.12)$$

Contoh 10.4: Dengan menggunakan Maltab dan aturan Trapezoid, tentukan nilai integral dari persamaan (10.13) berikut

$$\int_0^{\eth} \sin(x)\, dx \quad \ldots\ldots\ldots \quad (10.13)$$

Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (10.13) dengan aturan Trapezoid sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Integral Numerik Dengan Aturan Trapezoid
a = 0;
b = pi;
n = 11;
h = (b-a)/(n-1);
x = linspace(a,b,n);
f = sin(x);
%
I_trap   = (h/2)*(f(1)+ 2*sum(f(2:n-1)) + f(n))
err_mid = 2 - I_trap
```

Hasil program

```
I_trap =
    1.9835

err_mid =
    0.0165
```

Hasil perhitungan dengan menggunakan Matlab diperoleh hasil perhitungan integral numerik persamaan (10.13) sebesar 1.9835 dengan tingkat kesalahan sebesar 0.0082. Selain itu persamaan (10.13) dapat juga diselesaikan dengan menggunakan fungsi yang ada di Matlab. Adapun fungsi tersebut adalah fungsi ***trapz*** dan contoh penggunaan fungsi *trapz* ini diperlihatkan pada contoh 10.5 berikut

Contoh 10.5: Dengan menggunakan Maltab dan aturan Trapezoid, tentukan nilai integral dari persamaan (10.14) berikut

$$\int_0^{\pi} \sin(x)\, dx \quad \text{.......} \quad (10.14)$$

Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (10.14) dengan aturan Trapezoid sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Integral Numerik Dengan Fungsi Trapz
a = 0;
b = pi;
n = 11;
h = (b-a)/(n-1);
x = linspace(a,b,n);
f = sin(x);
%
I_trapz   = trapz(x,f)
```

Hasil program

```
I_trapz =
    1.9835
```

Hasil perhitungan dengan menggunakan Matlab diperoleh hasil perhitungan integral numerik persamaan (10.14) sebesar 1.9835.

10.5 Integral Numerik Dengan Aturan Simpson

Integral numerik dengan aturan Simpson dinyatakan dalam bentuk persamaan (10.15) dan (10.16) berikut

$$\int_a^b f(x)\, dx \approx \frac{h}{3}\left[f(x_i) + 4\left(\sum_{i=1, i\ \text{odd}}^{n-1} f(x_i) \right) + 2\left(\sum_{i=2, i\ \text{even}}^{n-2} f(x_i) + f(x_n) \right) \right] \quad \text{......} \quad (10.15)$$

$$\int_{x_i}^{x_{i+1}} f(x)\, dx = \frac{h}{3}\left(f(x_{i-1}) + 4f(x_i) + f(x_{i+1}) \right) + O(h^5) \quad \text{...........} \quad (10.16)$$

Contoh 10.6: Dengan menggunakan Maltab dan aturan Simpson, tentukan nilai integral dari persamaan (10.17) berikut

$$\int_0^{\pi} \sin(x)\, dx \quad \text{...} \quad (10.17)$$

Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (10.17) dengan aturan Simpson sebagai berikut

```
clc
clear all
close all
close all hidden
%
% Integral Numerik Dengan Aturan Simpson
a = 0;
b = pi;
n = 11;
h = (b-a)/(n-1);
x = linspace(a,b,n);
f = sin(x);
%
I_simp   = (h/3)*(f(1)+ 2*sum(f(1:2:n-1)) + 4*sum(f(2:2:n-1))
+ f(n))
err_mid = 2 - I_simp
```

Hasil program

```
I_simp =
    2.0001

err_mid =
  -1.0952e-04
```

Hasil perhitungan dengan menggunakan Matlab diperoleh hasil perhitungan integral numerik persamaan (10.17) sebesar 2.0001 dengan tingkat kesalahan sebesar 0.0001095

10.6 Rangkuman

Integral numerik adalah suatu cara untuk menghitung aproksimasi luas daerah di bawah fungsi yang dimaksud pada selang yang diberikan. Perhitungan integral numerik dilakukan dengan aturan Riemann kiri, aturan Riemann kanan, aturan titik tengah, aturan trapezoid dan aturan simpson. Masing - masing aturan akan memberikan tingkat akurasi yang berbeda – beda tergantung fungsi pendekatan yang digunakan.

10.7 Soal - Soal

Soal 10.1: Dengan menggunakan Maltab dan aturan Riemann kiri, tentukan nilai integral dari persamaan (10.18) berikut

$$\int_{0}^{ð/4} x^2 \sin(x)\, dx \qquad \text{.........} \qquad (10.18)$$

Soal 10.2: Dengan menggunakan Maltab dan aturan Riemann kanan, tentukan nilai integral dari persamaan (10.19) berikut

$$\int_{1.00}^{1.50} x^2 \ln(x)\, dx \qquad \text{.........} \qquad (10.19)$$

Soal 10.3: Dengan menggunakan Maltab dan aturan Titik Tengah, tentukan nilai integral dari persamaan (10.20) berikut

$$\int_{3.00}^{3.50} \frac{x}{\sqrt{x^2-4}}\, dx \qquad \text{.........} \qquad (10.20)$$

Soal 10.4: Dengan menggunakan Maltab dan aturan Trapezoid, tentukan nilai integral dari persamaan (10.21) berikut

$$\int_{0.00}^{1.00} \sqrt{1+x^2}\, dx \qquad \text{.........} \qquad (10.21)$$

Soal 10.5: Dengan menggunakan Maltab dan aturan Simpson, tentukan nilai integral dari persamaan (10.22) berikut

$$\frac{1}{ð}\int_{0}^{ð} \cos(\sin(x))\, dx \qquad \text{.........} \qquad (10.22)$$

BAB XI
TURUNAN NUMERIK

11.1 Pendahuluan

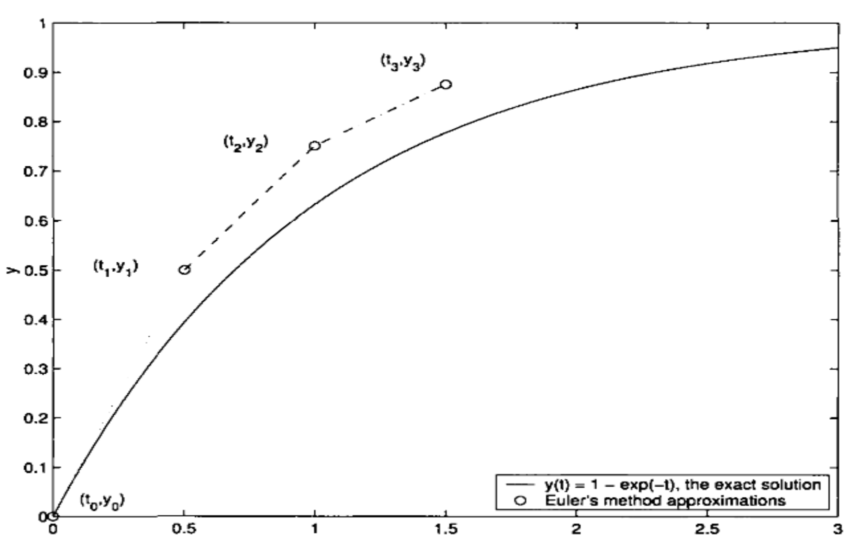
Pada bagian ini dibahas tentang turunan numerik. Turunan numerik adalah penentuan nilai hampiran turuan fungsi f. Meskipun metoda numerik untuk menghitung turunan fungsi tersedia tetapi perhitungan turunan sedapat mungkin dihindari. Alasannya nilai turuan numerik umumnya kurang teliti dibandingkan dengan nilai fungsinya. Dalam kenyataannya, turunan adalah limit dari hasil bagi selisih yaitu pengurangan dua buah nilai yang besar dan membaginya dengan bilangan yang kecil. Pembagian ini dapat menghasilkan turunan dengan galat yang besar. Beberapa metoda turunan numerik yang dibahas dalam bagian ini adlaah turunan numerik dengan metoda Euler dan turunan numerik dengan metoda Runge – Kutta.

11.2 Turunan Numerik Dengan Metoda Euler

Perhitungan turunan numerik dengan metoda Euler dilakukan dengan menggunakan persamaan (11.1) berikut

$$y_{n+1} = y_n + hf\left(t_n, y_n\right) \quad \text{dimana } n = 0, 1, 2, \ldots \qquad (11.1)$$

Secara Geometri metoda Euler ini dapat diilustrasikan dengan Gambar 11.1 berikut



Gambar 11.1 Ilustrasi Metoda Euler Secara Geometris

Contoh 11.1: Dengan menggunakan Maltab dan metoda Euler, hitung turunan numerik dari persamaan (11.2) berikut

$$\frac{dy}{dt} = -0.1000\left(y - 10.0000\right) \quad \text{.......} \quad (11.2)$$

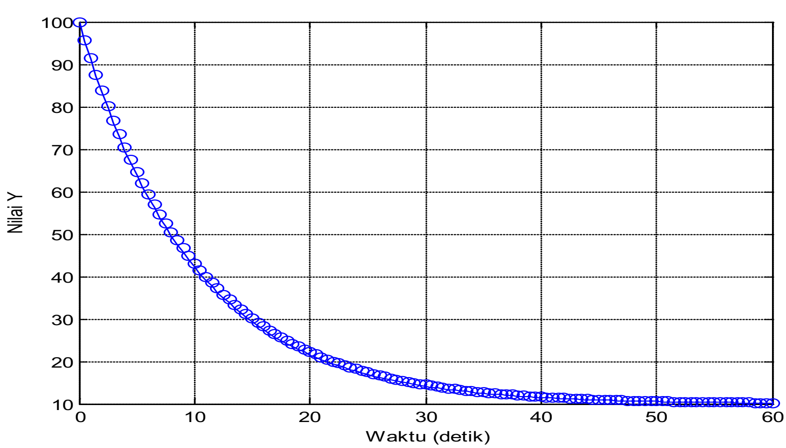
Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (11.2) dengan metoda Euler sebagai berikut

```
clc
clear all
close all
close all hidden
%
[t,y] = feuler('f501',[0 60],100,0.5);
plot(t,y)
grid on
xlabel(' Waktu (detik)')
ylabel(' Nilai Y')
hold on
plot(t,90*exp(-0.1000*t)+10.0000,'o');
hold off;
```

Hasil program

Berdasarkan hasil program diperoleh solusi dari persamaan (11.2) yang diperlihatkan pada Gambar 11.2 berikut



Gambar 11.2 Grafik Solusi Persamaan (11.2)

Berdasarkan Gambar 11.2 terlihat bahwa solusi dari persamaan (11.2) dengan metoda Euler sama dengan solusi eksak.

Kode Matlab untuk metoda Euler sebagai berikut

```
function [tvals, yvals]=feuler(f,tspan, startval,step)
steps=(tspan(2)-tspan(1))/step+1;
y=startval;t=tspan(1);
yvals=startval;tvals=tspan(1);
for i=2:steps
  y1=y+step*feval(f,t,y); t1=t+step;
  %collect values together for output
  tvals=[tvals, t1]; yvals=[yvals, y1];
  t=t1;y=y1;
end
```

Kode Matlab untuk persamaan (11.2) sebagai berikut

```
function yprime=f501(t,y)
yprime=-0.1*(y-10);
```

11.3 Turunan Numerik Dengan Metoda Runge- Kutta

Perhitungan turunan numerik dengan metoda Runge – Kutta. Metoda Runge Kutta ini mempuyai beberapa bentuk variasi diantaranya metoda Runge – Kutta klasik, metoda Runge – Kutta - Merson dan metoda Butcher – Rugge – Kutta. Secara umum metoda Runge – Kutta ini mempuyai bentuk persamaan yang dinyatakan dalam bentuk persamaan (11.3) s/d (11.5) berikut

$$k_1 = hf\left(t_n, y_n\right) \quad \text{.........} \quad (11.3)$$

$$k_i = hf\left(t_n + hd_i\ , y_n + \sum_{j=1}^{i-1} c_{ij}k_j\right) \quad \text{.........} \quad (11.4)$$

$$y_{n+1} = y_n + \sum_{j=1}^{p} b_j k_j \quad \text{.........} \quad (11.5)$$

Contoh 11.2: Dengan menggunakan Maltab dan metoda Runge - Kutta klasik , hitung turunan numerik dari persamaan (11.6) berikut

$$\frac{dy}{dt} = -y \quad \text{.........} \quad (11.6)$$

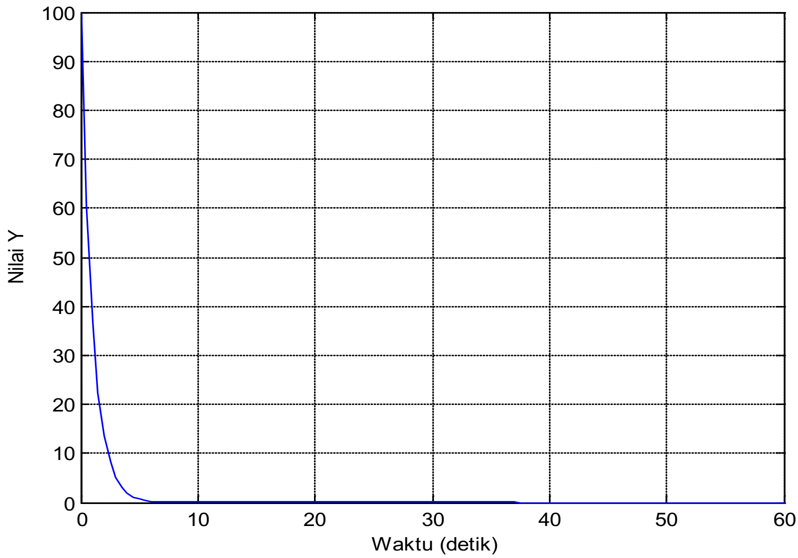
Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (11.6) dengan metoda Runge – Kutta klasik sebagai berikut

```
clc
clear all
close all
close all hidden
%
[t,y] = rkgen('f502',[0 60],100,0.5,1);
plot(t,y)
grid on
xlabel(' Waktu (detik)')
ylabel(' Nilai Y')
```

Hasil program

Berdasarkan hasil program diperoleh solusi dari persamaan (11.6) yang diperlihatkan pada Gambar 11.3 berikut



Gambar 11.3 Grafik Solusi Persamaan (11.3)

Contoh 11.3: Dengan menggunakan Maltab dan metoda Runge - Kutta - Butcher, hitung turunan numerik dari persamaan (11.7) berikut

$$\frac{dy}{dt} = -y \qquad \text{(11.7)}$$

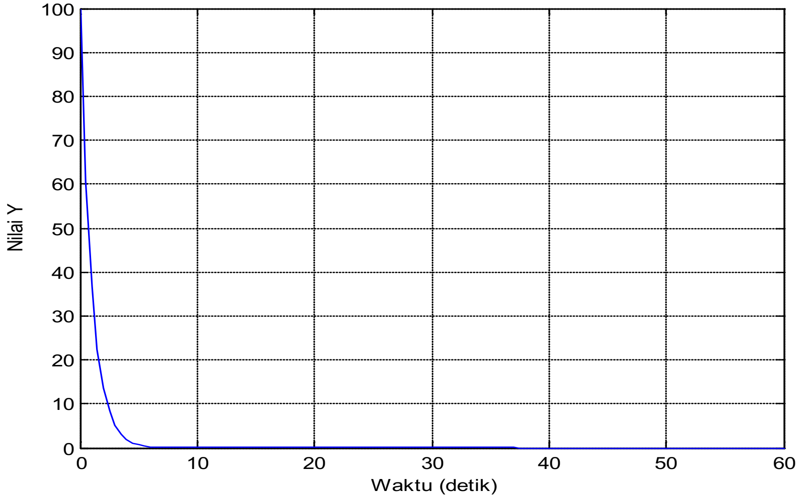
Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (11.7) dengan metoda Runge – Kutta - Butcher sebagai berikut

```
clc
clear all
close all
close all hidden
%
[t,y] = rkgen('f502',[0 60],100,0.5,2);
plot(t,y)
grid on
xlabel(' Waktu (detik)')
ylabel(' Nilai Y')
```

Hasil program

Berdasarkan hasil program diperoleh solusi dari persamaan (11.7) yang diperlihatkan pada Gambar 11.4 berikut



Gambar 11.4 Grafik Solusi Persamaan (11.4)

Contoh 11.3: Dengan menggunakan Maltab dan metoda Runge - Kutta - Merson , hitung turunan numerik dari persamaan (11.8) berikut

$$\frac{dy}{dt} = -y \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (11.8)$$

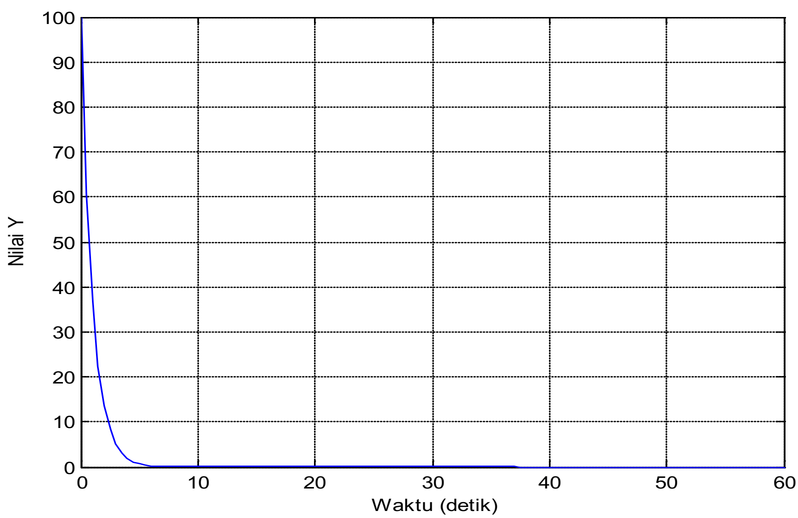
Jawab :

Adapun kode Matlab untuk penyelesaian persamaan (11.8) dengan metoda Runge – Kutta - Merson sebagai berikut

```
clc
clear all
close all
close all hidden
%
[t,y] = rkgen('f502',[0 60],100,0.5,2);
plot(t,y)
grid on
xlabel(' Waktu (detik)')
ylabel(' Nilai Y')
```

Hasil program

Berdasarkan hasil program diperoleh solusi dari persamaan (11.8) yang diperlihatkan pada Gambar 11.5 berikut



Gambar 11.5 Grafik Solusi Persamaan (11.4)

Kode Matlab untuk metoda Runge – Kutta Klasik, metoda Runge – Kutta Butcher dan metoda Runge – Kutta – Merson sebagai berikut

```
function[tvals,yvals]=rkgen(f,tspan,startval,step,method)
b=[ ];c=[ ];d=[ ];
if method <1 | method >3
  disp('Method number unknown so using Classical');
  method=1;
end;
if method==1
  order=4;
  b=[ 1/6 1/3 1/3 1/6]; d=[0 .5 .5 1];
  c=[0 0 0 0;0.5 0 0 0;0 .5 0 0;0 0 1 0];
  disp('Classical method selected');
elseif method ==2
  order=6;
  b=[0.07777777778 0 0.355555556 0.13333333 ...
     0.355555556 0.0777777778];
  d=[0 .25 .25 .5 .75 1];
  c(1:4,:)=[0 0 0 0 0 0;0.25 0 0 0 0 0;0.125 0.125 0 0 0 0; ...
            0 -0.5 1 0 0 0];
  c(5,:)=[.1875 0 0 0.5625 0 0];
  c(6,:)=[-.4285714 0.2857143 1.714286 -1.714286 1.1428571 0];
  disp('Butcher method selected');
else
  order=5;
  b=[1/6 0 0 2/3 1/6];
  d=[0 1/3 1/3 1/2 1];
  c=[0 0 0 0 0;1/3 0 0 0 0;1/6 1/6 0 0 0;1/8 0 3/8 0 0; ...
     1/2 0 -3/2 2 0];
  disp('Merson method selected');
end;
steps=(tspan(2)-tspan(1))/step+1;
y=startval; t=tspan(1);
yvals=startval; tvals=tspan(1);
for j=2:steps
  k(1)=step*feval(f,t,y);
  for i=2:order
    k(i)=step*feval(f,t+step*d(i),y+c(i,1:i-1)*k(1:i-1)');
  end;
  y1=y+b*k'; t1=t+step;
  %collect values together for output
  tvals=[tvals, t1]; yvals=[yvals, y1];
  t=t1; y=y1;
end;
```

Kode Matlab untuk persamaan (11.6) s/d (11.8) sebagai berikut

```
function yprime=f502(t,y)
yprime=-y;
```

11.4 Rangkuman

Turunan numerik adalah adalah penentuan nilai hampiran turuan fungsi f. Perhitungan turuan numerik dilakukan dengan menggunakan metoda Euler, metoda metoda Runge – Kutta klasik, metoda Runge – Kutta - Merson dan metoda Butcher – Rugge – Kutta. Masing – masin metoda ini memberikan tingkat akurasi yang berbeda – beda tergantung metoda yang digunakan.

11.5 Soal - Soal

Soal 11.1: Dengan menggunakan Maltab dan metoda Euler, hitung turunan numerik dari persamaan (11.9) berikut

$$\frac{dy}{dt} = -0.0500y \quad \text{.........} \quad (11.9)$$

Untuk t = 0 s.d t = 15, y = 50 dan h = 0.1000

Soal 11.2: Dengan menggunakan Maltab dan metoda Runge – Kutta - Klasik, hitung turunan numerik dari persamaan (11.10) berikut

$$\frac{dy}{dt} = -0.0500y \quad \text{.........} \quad (11.10)$$

Untuk t =0 s.d t = 15, y = 50 dan h= 0.0100

Soal 11.3: Dengan menggunakan Maltab dan metoda Runge - Kutta - Butcher,, hitung turunan numerik dari persamaan (11.11) berikut

$$\frac{dy}{dt} = -0.0500y \quad \text{.........} \quad (11.11)$$

Untuk t =0 s.d t = 15, y = 50 dan h= 0.0100

Soal 11.4: Dengan menggunakan Maltab dan metoda Runge - Kutta - Merson,, hitung turunan numerik dari persamaan (11.12) berikut

$$\frac{dy}{dt} = -0.0500y \quad \text{.........} \quad (11.12)$$

Untuk Untuk t =0 s.d t = 15, y = 50 dan h= 0.0100

DAFTAR PUSTAKA

1. Arhami , M. & Desiani , A., 2005. *Pemograman Matlab.* 1st ed. Jogjakarta: Andi Offset .
2. Away, G. A., 2006. *Matlab Programming.* 1st ed. Bandung : Informatika .
3. Chapra, S. C. & Canale , R. P., 2007. *Metode Numerik Untuk Teknik Dengan Penerapan Pada Komputer Pribadi.* Jakarta : Universitas Indonesia Press .
4. Kariadinata , R., 2013. *Aljabar Matriks Elementer.* 1st ed. Bandung : Pustaka Setia .
5. Kosasih, P. B., 2006 . *Komputasi Numerik Teori dan Aplikasi.* 1st ed. Jogjakarta: Andi Offset.
6. Linfield, G. & Penny, J., 1995. *Numerical Methods Using Matlab.* 1st ed. Maryland : Ellis Horwood.
7. Munir, R., 2010. *Metode Numerik Revisi Ketiga.* Bandung : Informatika .
8. Peranginangin, K., 2006. *Pengenalan Matlab.* 1st ed. Jogjakarta: Andi Offset.
9. Prasetyo, W. A., 2004. *Tips dan Trik Matlab (Vektorisasi, Optimisasi dan Mani[ulasi Array).* 1st ed. Jogjakarta: Andi Offset.
10. Pujiyanta, A., 2007. *Komputasi Numerik denga Matlab.* Jogjakarta: Graha Ilmu .
11. S., 2006. *Panduan Praktis Matlab.* 1st ed. Jogjakarta : Andi Offset.
12. S., 2014. *Komputasi Numerik (Pemograman Matlab Untuk Metoda Numerik).* 1st ed. Jogjakarta : Andi Offset.
13. Sianipar, R., 20013. *Pemograman Matlab Dalam Contoh dan Penerapan.* 1st ed. Bandung : Informatika .
14. Siauw, T. & Bayen , A. M., 2015. *An Introduction to Matlab Programming and Numerical Methods for Engineers.* 1st ed. San Diego : Elsevier.
15. Sutojo, T., B. N. & E. Z. d., 2010. *Teori dan Aplikasi Aljabar Linier dan Matriks.* 1st ed. Jogjakarta : Andi Offset.

INDEKS

L

M

P

R

S

Sekilas Tentang Penulis

Reri Afrianita, M.T, Saat ini bertugas sebagai salah seorang staff pengajar di Jurusan Teknik Lingkungan Fakultas Teknik Universitas Andalas Padang Sumatera Barat. Menyelesaikan program sarjana (S1) di Jurusan Teknik Sipil Fakultas Teknik Universitas Andalas tahun 2000 dan program pascasarjana (S2) di Departemen Teknik Lingkungan Institute Teknologi Bandung (ITB) dengan bidang keahlian manajemen lingkungan tahun 2005. Mata kuliah yang diajar antara lain Metode Numerik dan Dasar – Dasar Pemograman Komputer, Matematika Rekayasa, Mekanika Fluida 1 dan Mekanika Fluida 2

Heru Dibyo Laksono, M.T, Saat ini bertugas sebagai salah seorang staff pengajar di Jurusan Teknik Elektro Fakultas Teknik Universitas Andalas Padang Sumatera Barat. Menyelesaikan program sarjana (S1) di Jurusan Teknik Elektro Fakultas Teknik Universitas Andalas tahun 2000 dengan bidang keahlian sistem tenaga listrik dan program pascasarjana (S2) di Sekolah Tinggi Elektro dan Informatika (STEI) Institute Teknologi Bandung (ITB) dengan bidang keahlian sistem kendali tahun 2004. Selain akatif mengajar, penulis juga tela menghasilkan beberapa judul buku yang berkaitan dengan sistem kendali yang diterbitkan oleh beberapa penerbit skala nasional.